Technical Reference

Tektronix

TVS600 & TVS600A Series
Waveform Analyzers Command Reference

070-9917-00

Warning
The servicing instructions are for use by qualified personnel only. To avoid personal injury, do not perform any servicing unless you are qualified to do so. Refer to the Safety Summary prior to performing service.

This document supports firmware version 1.5 and above.

Copyright © Tektronix, Inc. All rights reserved. Licensed software products are owned by Tektronix or its suppliers and are protected by United States copyright laws and international treaty provisions.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, or subparagraphs (c)(1) and (2) of the Commercial Computer Software – Restricted Rights clause at FAR 52.227-19, as applicable.

Tektronix products are covered by U.S. and foreign patents, issued and pending. Information in this publication supercedes that in all previously published material. Specifications and price change privileges reserved.

Printed in the U.S.A.

Tektronix, Inc., P.O. Box 1000, Wilsonville, OR 97070–1000

TEKTRONIX and TEK are registered trademarks of Tektronix, Inc.

Table of Contents

Command Syntax

Commands

List of Figures

List of Tables

General Safety Summary

Review the following safety precautions to avoid injury and prevent damage to this product or any products connected to it. To avoid potential hazards, use this product only as specified.

Only qualified personnel should perform service procedures.

While using this product, you may need to access other parts of the system. Read the *General Safety Summary* in other system manuals for warnings and cautions related to operating the system.

To Avoid Fire or Personal Injury

Connect and Disconnect Properly. Do not connect or disconnect probes or test leads while they are connected to a voltage source.

Ground the Product. This product is grounded through the grounding conductor of the power cord. To avoid electric shock, the grounding conductor must be connected to earth ground. Before making connections to the input or output terminals of the product, ensure that the product is properly grounded.

Do not apply a potential to any terminal, including the common terminal, that exceeds the maximum rating of that terminal.

Replace Batteries Properly. Replace batteries only with the proper type and rating specified.

Do Not Operate Without Covers. Do not operate this product with covers or panels removed.

Use Proper Fuse. Use only the fuse type and rating specified for this product.

Avoid Exposed Circuitry. Do not touch exposed connections and components when power is present.

Wear Eye Protection. Wear eye protection if exposure to high-intensity rays or laser radiation exists.

Do Not Operate With Suspected Failures. If you suspect there is damage to this product, have it inspected by qualified service personnel.

Do Not Operate in Wet/Damp Conditions.

Do Not Operate in an Explosive Atmosphere.

Keep Product Surfaces Clean and Dry.

Provide Proper Ventilation. Refer to the manual's installation instructions for details on installing the product so it has proper ventilation.

Symbols and Terms

Terms in this Manual. These terms may appear in this manual:

WARNING. *Warning statements identify conditions or practices that could result in injury or loss of life.*

CAUTION. *Caution statements identify conditions or practices that could result in damage to this product or other property.*

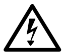
Terms on the Product. These terms may appear on the product:

DANGER indicates an injury hazard immediately accessible as you read the marking.

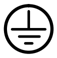
WARNING indicates an injury hazard not immediately accessible as you read the marking.

CAUTION indicates a hazard to property including the product.

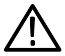
Symbols on the Product. The following symbols may appear on the product:



WARNING
High Voltage



Protective Ground
(Earth) Terminal



CAUTION
Refer to Manual

Double
Insulated

Preface

This manual is the command reference for the TVS600 & TVS600A Series Waveform Analyzers. These waveform analyzers are controlled through the use of SCPI (Standard Commands for Programmable Instruments) derived commands and IEEE 488.2 Common Commands. This manual describes how to use these commands to configure the waveform analyzer and access information generated by it or stored within it.

Related Manuals

This manual is part of a document set of standard accessory manuals and online documentation. This manual is the reference for thr SCPI command used by the waveform analyzer. The following documents support TVS600 and TVS600A operation and service:

Manual Name	Description
TVS600 & TVS600A Series Waveform Analyzers Reference	Provides an alphabetical listing of the programming commands. It is the quick command reference and is a standard accessory.
TVS600 & TVS600A Series Waveform Analyzers Users Manual	Describes installation of and the features of the waveform analyzer. It is a standard accessory.
TVS600A Online SFP Help for the VXI*plug&play* Soft Front Panel	Documents the Soft Front Panel, an application that ships with this product. The TVS600A VXI*plug&play* software is a standard accessory included with the waveform analyzer product.
TVS600A Online Driver Help for the VXI*plug&play* Driver	Documents the robust library of functions that ships with this product. The TVS600A VXI*plug&play* software is a standard accessory included with this user manual.
TVS600 & TVS600A Series Waveform Analyzers Service Manual	Describes how to service the instrument to the module level. This optional manual must be ordered separately.

- The *TVS600 Series Waveform Analyzers Reference* (Tektronix part number 070-9284-XX) provides an alphabetical listing of the programming commands. This manual is a standard accessory.
- The *TVS600 Series Waveform Analyzers Service Manual* (Tektronix part number 070-9285-XX) describes how to service the instrument to the module level. This optional manual must be ordered separately.

Default Model

This manual documents the TVS621, TVS621A, TVS625, TVS625A, TVS641, TVS641A, TVS645, and TVS645A waveform analyzers. Take note of the following conventions used when referencing these products:

- Generally, the name "TVS600A" (or just "waveform analyzer") is used when providing information common to the TVS600 and TVS600A series of waveform analyzers.
- The labels "TVS600 only" and "TVS600A only" are used when providing information that pertains only to those models.
- The more specific names, listed above, are used when providing information that pertains only to a specific model, such as the TVS625A.

Command Syntax

Command Syntax

This section contains information on the Standard Commands for Programmable Instruments (SCPI) and IEEE 488.2 Common Commands you can use to program your waveform analyzer.

SCPI Commands and Queries

SCPI is a standard created by a consortium that provides guidelines for remote programming of instruments. These guidelines provide a consistent programming environment for instrument control and data transfer. This environment uses defined programming messages, instrument responses, and data format across all SCPI instruments, regardless of manufacturer. The waveform analyzer uses a command language based on the SCPI standard.

The SCPI language is based on a hierarchical or tree structure (see Figure 1–1) that represents a subsystem. The top level of the tree is the root node; it is followed by one or more lower-level nodes.

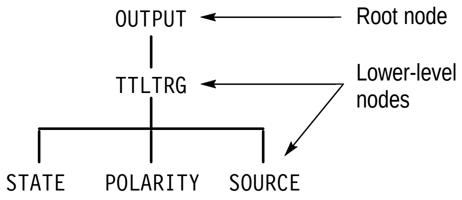


Figure 1–1: Example of SCPI subsystem hierarchy tree

You can create commands and queries from these subsystem hierarchy trees. Commands specify actions for the instrument to perform. Queries return measurement data and information about parameter settings.

Creating Commands

SCPI commands are created by stringing together the nodes of a subsystem hierarchy and separating each node by a colon.

In Figure 1–1, OUTPUT is the root node and TTLTRG, STATE, POLARITY, and SOURCE are lower-level nodes. To create a SCPI command, start with the root node OUTPUT and move down the tree structure adding nodes until you reach the end of a branch. Most commands and some queries have parameters; you must include a value for these parameters. If you specify a parameter value that is out of range, the parameter will be generally set to a default value. The command descriptions, which start on page 2–1, list the valid values for all parameters.

For example, OUTPUT:TTLTRG1:STATE ON is a valid SCPI command created from the hierarchy tree in Figure 1–1.

Creating Queries

To create a query, start at the root node of a tree structure, move down to the end of a branch, and add a question mark. OUTPUT:TTLTrg:STATe? is an example of a valid SCPI query using the hierarchy tree in Figure 1–1.

Parameter Types

Every parameter in the command and query descriptions is of a specified type. The parameters are enclosed in brackets, such as <pattern>. The parameter type is listed after the parameter and is enclosed in parentheses, for example, (discrete). Some parameter types are defined specifically for the waveform analyzer command set and some are defined by ANSI/IEEE 488.2-1992 (see Table 1–1).

Table 1–1: Parameter types used in syntax descriptions

Parameter Type	Description	Example
binary	Binary numbers	#B0110
binary block[1]	A specified length of binary data	#512234xxxxx . . . where 5 indicates that the following 5 digits (12234) specify the length of the data in bytes; xxxxx ... indicates the binary data
boolean	Boolean numbers or values	ON or 1 OFF or 0
discrete	A list of specific values	HIGH, LOW, MID, PRBS23
hexadecimal[2]	Hexadecimal numbers (0–9, A, B, C, D, E, F)	#HAA, #H1
NR1[2,3] numeric	Integers	0, 1, 15, –1
NR2[2] numeric	Decimal numbers	1.2, 3.141516, –6.5
NR3[2] numeric	Floating point numbers	3.1415E–9, –16.1E5
NRf[2] numeric	Flexible decimal number that may be type NR1, NR2 or NR3	See NR1, NR2, NR3 examples
string[4]	Alphanumeric characters (must be within quotation marks)	“Testing 1, 2, 3”

[1] **Defined in ANSI/IEEE 488.2 as “Definite Length Arbitrary Block Response Data.”**

[2] **An ANSI/IEEE 488.2–1992-defined parameter type.**

[3] **Some commands and queries will accept a hexadecimal value even though the parameter type is defined as NR1.**

[4] **Defined in ANSI/IEEE 488.2 as “String Response Data.”**

Abbreviating Commands, Queries, and Parameters

You can abbreviate most SCPI commands, queries, and parameters to an accepted short form. This manual shows these short forms as a combination of upper and lower case letters. The upper case letters indicate the accepted short form of a command. As shown in Figure 1–2, you can create a short form by using only the upper case letters. The accepted short form and the long form are equivalent and request the same action of the instrument.

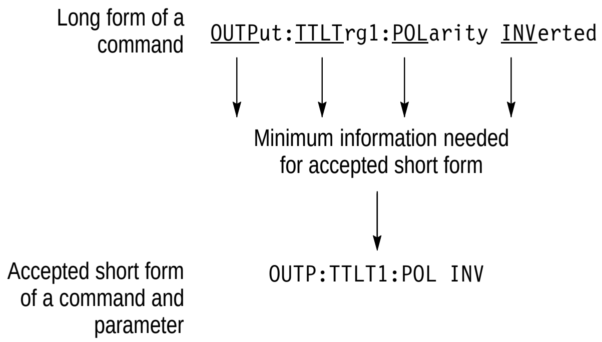
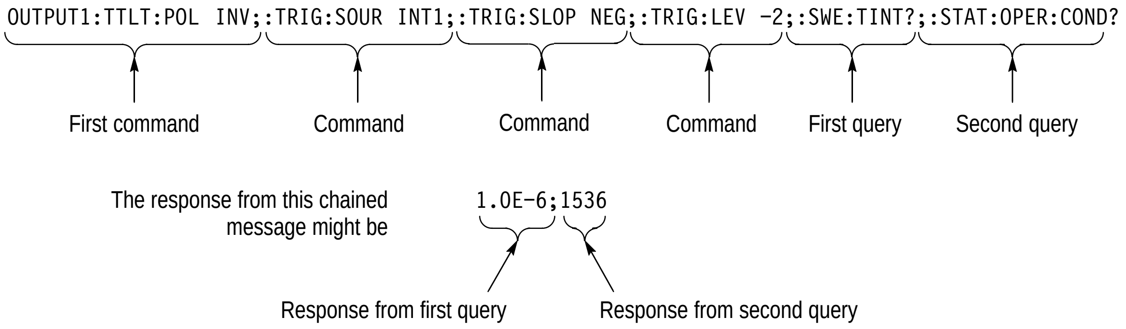

Long form of a command
OUTPut:TTLTrg1:POLarity INVerted
Minimum information needed for accepted short form
Accepted short form of a command and parameter
OUTP:TTLT1:POL INV

Figure 1–2: Example of abbreviating a command

***NOTE**. The numeric part of a command or query must always be included in the accepted short form. In Figure 1–2, the "1" of "TTLTRG1" is always included in the command or query.*

Chaining Commands and Queries

You can chain several commands or queries together into a single message. To create a chained message, first create a command or query, add a semicolon (;), and then add more commands or queries and semicolons until you are done. If the command following a semicolon is a root node, precede it with a colon (:). Figure 1–3 illustrates a chained message consisting of several commands and queries. A semicolon is not required after the final end or query in a chained message. Responses to any queries in your message are separated by semicolons.



```
OUTPUT1:TTLT:POL INV;:TRIG:SOUR INT1;:TRIG:SLOP NEG;:TRIG:LEV -2;:SWE:TINT?;:STAT:OPER:COND?
```

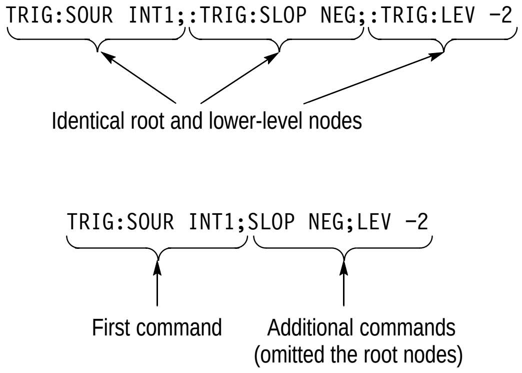
First command
Command
Command
Command
First query
Second query
The response from this chained message might be

```
1.0E-6;1536
```

Response from first query
Response from second query

Figure 1–3: Example of chaining commands and queries

If a command or query has the same root and lower-level nodes as the previous command or query, you can omit these nodes. In Figure 1–4, the second command has the same root node (TRIG) as the first command, so these nodes can be omitted.



```
TRIG:SOUR INT1;:TRIG:SLOP NEG;:TRIG:LEV -2
```

Identical root and lower-level nodes

```
TRIG:SOUR INT1;SLOP NEG;LEV -2
```

First command
Additional commands (omitted the root nodes)

Figure 1–4: Example of omitting root and lower-level nodes in chained message

General Rules

Here are some general rules for using SCPI commands, queries, and parameters:

- You can use single (‘ ’) or double (“ ”) quotation marks for quoted strings, but you cannot use both types of quotation marks for the same string.

correct:	“This string uses quotation marks correctly.”
correct:	‘This string also uses quotation marks correctly.’
incorrect:	“This string does not use quotation marks correctly.’

- You can use upper case, lower case, or a mixture of both cases for all commands, queries, and parameters.

  ```
  OUTPUT1:TTLTRG:POLARITY INVERTED
  ```

  is the same as

  ```
  output1:ttltrg:polarity inverted
  ```

  and

  ```
  OUTPUT1:ttltrg:polarity INVERTED
  ```

- No embedded spaces are allowed between or within nodes.

  correct: `OUTPUT1:TTLTRG:POLARITY INVERTED`

  incorrect: `OUTPUT1: TTLTRG: POLARITY INV ERTED`

IEEE 488.2 Common Commands

Description

ANSI/IEEE Standard 488.2 defines the codes, formats, protocols, and usage of common commands and queries used on the interface between the controller and the instruments. The waveform analyzer complies with this standard.

Command and Query Structure

The syntax for an IEEE 488.2 common command is an asterisk (*) followed by a command and, optionally, a space and parameter value. The syntax for an IEEE 488.2 common query is an asterisk (*) followed by a query and a question mark. All of the common commands and queries are listed in the last part of the *Syntax and Commands* section. The following are examples of common commands:

- `*ESE 16`
- `*CLS`

The following are examples of common queries:

- `*ESR?`
- `*IDN?`

Backus-Naur Form Definition

This manual may describe commands and queries using the Backus-Naur Form (BNF) notation. Table 1–2 defines the standard BNF symbols:

Table 1–2: BNF symbols and meanings

Symbol	Meaning
< >	Defined element
::=	Is defined as
|	Exclusive OR
{ }	Group; one element is required
[ ]	Optional; can be omitted
. . .	Previous element(s) may be repeated
( )	Comment

Message Terminators

This manual uses `<EOM>` (End of message) to represent a message terminator.

Symbol	Meaning
<EOM>	Message terminator

The end-of-message terminator may be the END message (EOI asserted concurrently with the last data byte), the ASCII code for line feed (LF) sent as the last data byte, or both. The waveform analyzer always terminates messages with LF and EOI. It allows white space before the terminator.

Constructed Mnemonics

Some header mnemonics specify one of a range of mnemonics. For example, a channel mnemonic can be either `INP1`, `INP2`, `INP3`, or `INP4`. You use these mnemonics in the command just as you do any other mnemonic. For example, there is a `INP1:FILT` command, and there is also an `INP2:FILT` command. In the command descriptions, this list of choices is abbreviated as `INP<n>`.

Block Arguments

Several waveform analyzer commands use a block argument form:

Symbol	Meaning
`<NZDig>`	A non-zero digit character, in the range 1–9
`<Dig>`	A digit character, in the range 0–9
`<DChar>`	A character with the hex equivalent of 00 through FF hexadecimal (0 through 255 decimal)

`<Block>`	A block of data bytes, defined as:
	`<Block> ::=`
	`{ #<NZDig><Dig>[<Dig>...][<DChar>...]`
	`  | #0[<DChar>...]<terminator> }`

`<NZDig>` specifies the number of `<Dig>` elements that follow. Taken together, the `<Dig>` elements form a decimal integer that specifies how many `<DChar>` elements follow. The #0 format is for blocks of indifinite length and an END message termination is required.

Commands

Commands

This section describes each command and query in the waveform analyzer. The commands are organized by subsystem groups and the commands in each group are in alphabetical order. In Figure 2–1, each block is a root node and the commands within a block are subsystems. For example, SENSe is a root node and AVERage is a subsystem of the SENSe node.

Overview

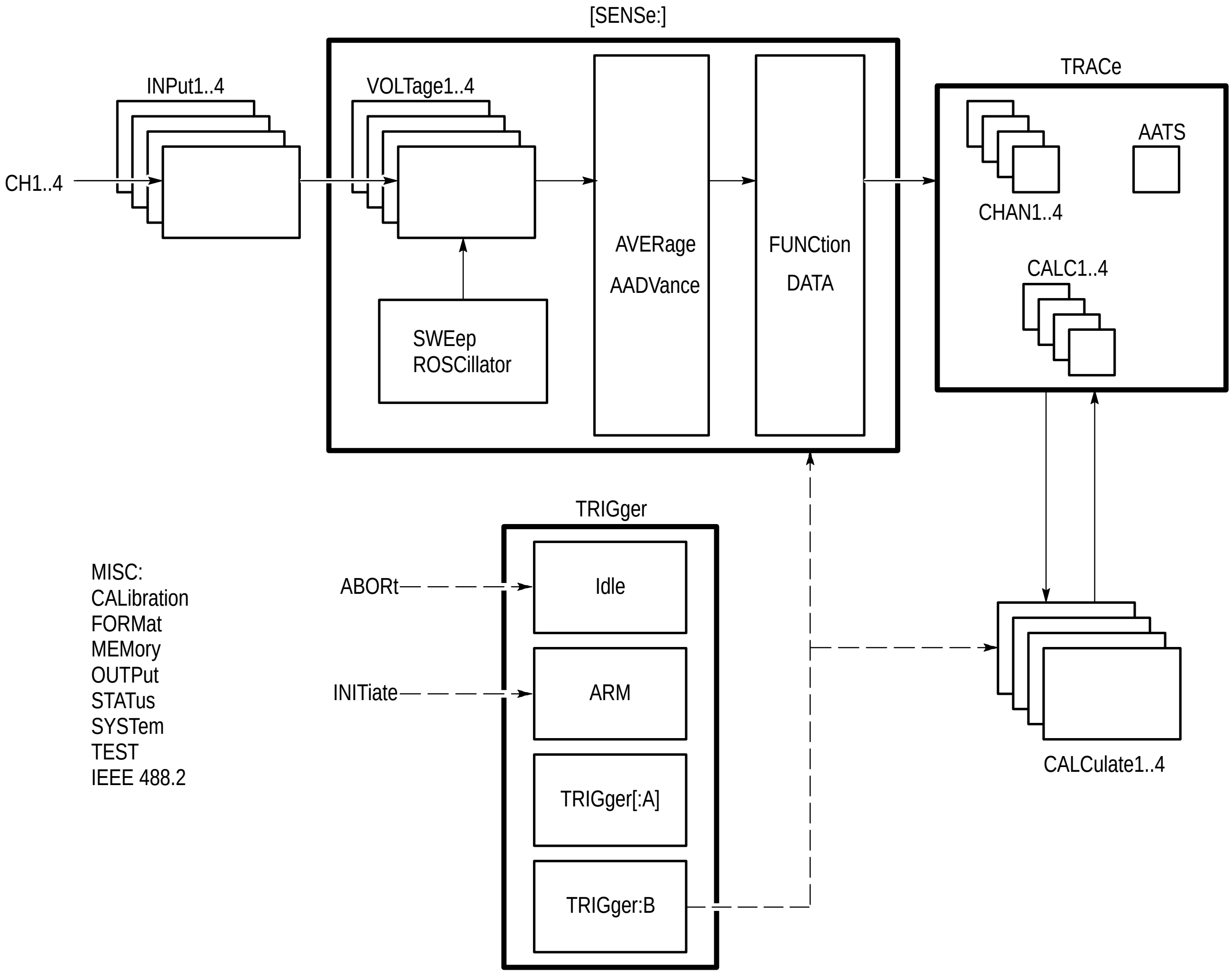


Figure 2–1: Instrument model showing root-level nodes and subsystems

AADVance Subsystem

This section describes the commands in the [SENSe:]AADVance subsystem. See Figure 2–2. These commands control how auto-advance acquisition records are acquired and transferred to a VXIbus controller. A functional model of the AADVance subsystem is shown in Figure 2–3.

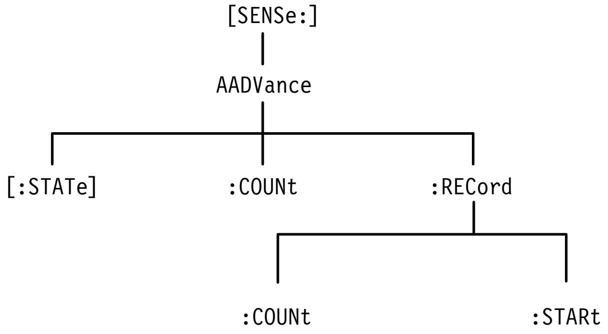


Figure 2–2: AADVance subsystem hierarchy

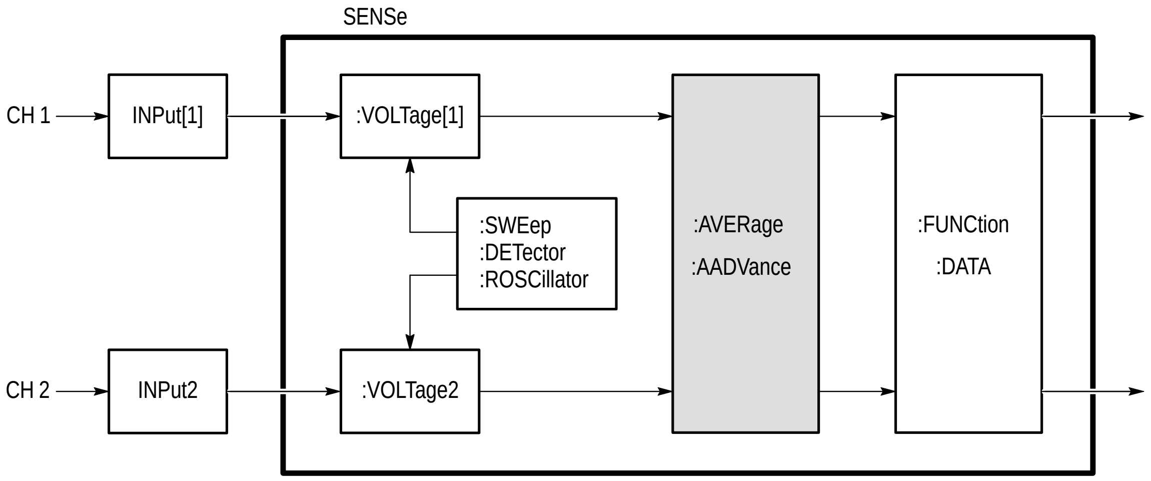


Figure 2–3: AADVance subsystem functional model

AADVance
AADVance?

Sets or queries the state of the auto-advance acquisition mode. In the auto-advance mode the waveform analyzer acquires a sequence of data records for each active channel. The delay between one acquisition record and the next one is very short and is due only to the minimal re-arm time and any trigger holdoff you set. Use the command AADVance:COUNt to set the number of records to acquire. You cannot use acquisition modes average, envelope, or peak detect (set using [SENSe:]AVERage command) when using the auto-advance acquisition mode.

The auto-advance mode of acquisition affects all enabled channels (XTIM:VOLT <n>). You cannot acquire one channel in the auto-advance mode and acquire another with the normal acquisition mode.

Syntax

```
[SENSe:]AADVance[:STATe] <boolean>
[SENSe:]AADVance[:STATe]?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NF1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

Enabling auto-advance acquisition disables [SENSe:]AVERage acquisition, ENVelope or SCALar type.

Examples

Command: `AADV ON`

Query: `AADV?`

Response: `1`

Related Commands

AADVance:COUNt
AVERage

AADVance:COUNt AADVance:COUNt?

Sets or queries the number of records to acquire in the auto-advance acquisition mode. The maximum number of Auto-advance records that you can acquire depends on the record length and the number of active channels. A setting of zero (0) acquires enough records to fill the DSP memory, regardless of the current acquisition system settings. MAXimum acquires enough records to fill DSP memory based on the current acquisition settings. The distinction is that with a change such as the number of active channels, the MAXimum COUNt setting is not adjusted, whereas the zero COUNt setting adjusts to ensure memory is just filled within the new conditions.

To determine the current value for MAX, first set all acquisition parameters. Set AADV:COUN to MAX. Finally, query with AADV:COUN? to return the current value for MAX.

Syntax

```
[SENSe:]AADVance:COUNt <count>
[SENSe:]AADVance:COUNt?
```

Parameters

<count>		Query response
<NRf>		<NR1>
0	(Fill acquisition memory)	
1 to MAX	(Acquire number)	
MINimum		1
MAXimum		<depends on configuration>

Reset Value

1

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set count to an illegal value.

Dependencies

None

Examples

Command: `AADV:COUN 100`

Query: `AADV:COUN?`

Response: `100`

Related Commands

AADVance

AADVance:RECord:COUNt
AADVance:RECord:COUNt?

Sets or queries the number of auto-advance acquisition records to transfer in response to the commands DATA?, TRACe?, TRACe:COPY?, or TRACe:LIST?. The maximum :COUNt value depends on the number of acquired records and the value of AADVance:RECord:STARt. A setting of zero (0) selects all acquisition records, beginning with :STARt, for transfer regardless of the current acquisition system settings. MAXimum selects all acquisition records for transfer based on the current acquisition settings. If you change a setting, such as the number of active channels, the MAXimum setting will not be adjusted, but a setting of zero will automatically adjust for the new acquisition settings.

Syntax

```
[SENSe:]AADVance:RECord:COUNt <count>
[SENSe:]AADVance:RECord:COUNt?
```

Parameters

<count>		Query response
<NRf>		<NR1>
0	(Transfer all records)	
1 to MAX	(Transfer number of records)	
MINimum		1
MAXimum		<depends on configuration>

Reset Value

1

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set count to an illegal value.

Dependencies

None

Examples

Command: `AADV:REC:COUN 100`

Query: `AADV:REC:COUN?`

Response: `100`

Related Commands

AADVance
AADVance:COUNt
AADVance:RECord:STARt
CALC:AAML

AADVance:RECord:STARt
AADVance:RECord:STARt?

Sets or queries the number of the first auto-advance waveform record to transfer in response to the commands DATA?, TRACe?, and TRACe:COPY?. The maximum :STARt value depends on the number of acquired records. A setting of zero (0) selects the last waveform record for transfer. Negative settings select waveform records referenced from the last one, the zero record. For example, the –1 record is the second from last and the –2 record is third from last.

Note that auto-advance acquisition always starts with record number one.

Syntax

```
[SENSe:]AADVance:RECord:STARt <start>
[SENSe:]AADVance:RECord:STARt?
```

Parameters

<start>		Query response
<NRf>		<NR1>
–MAX + 1 to –1	(Number before last record)	
0	(Last record)	
1 to MAX	(Number from first record)	
MINimum		1
MAXimum		<depends on configuration>

Reset Value

1

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set start to an illegal value.

Dependencies

None

Examples

Command: `AADV:REC:STAR 100`

Query: `AADV:REC:STAR?`

Response: `100`

Related Commands

AADVance
AADVance:COUNt
AADVance:RECord:COUNt
CALC:AAML

ARM Subsystem

This section describes the commands in the ARM subsystem. See Figure 2–4. These commands operate with the TRIGger, INITiate, and ABORt subsystems to trigger acquisitions.

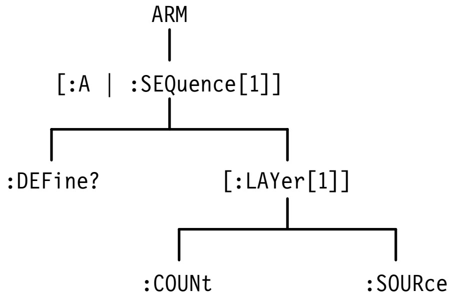


Figure 2–4: ARM subsystem hierarchy

ARM:DEFine? (Query Only)

Returns the predefined SEQuence1 alias. :A is a pre-defined alias for :SEQuence[1]. The commands ARM:DEFine? and TRIGger:DEFine? are aliases which produce the same result.

Syntax

```
ARM[:SEQuence[1]]:DEFine?
```

Parameters

<sequence_alias>	Query response
Not applicable	<string> “A”

Reset Value

“A”

Errors and Events

None

Dependencies

None

Examples

Query: `ARM:DEF?`

Response: `"A"`

Related Commands

TRIGger:DEFine?
TRIGger:SEQuence2:DEFine?

ARM:SOURce
ARM:SOURce?

Sets or queries the source that will arm the acquisition system. You can specify only one source at a time and it is shared by all acquired channels. Setting the arm source to BUS configures the event detector to accept and arm on either the *TRG or the VXIbus word serial <Trigger> command. The ECLTrg and TTLTrg sources provide normal and inverted access to the standard ECLT and TTLT signals on the VXI P2 bus. TTLTrg<n> arms when the TTLT line is low. ITTLTrg<n> arms when the TTLT line is high. The ECLTrg<n> arms when the ECLT line is high. IECLTrg<n> arms when the ECLT line is low. EXT arms when the EXT input is a TTL low.

EXTernal is the front-panel BNC connector labeled Arm Input. Setting the arm source to IMMediate bypasses event detection and immediately arms the acquisition system.

Syntax

```
ARM[:A][:LAYer[1]]:SOURce <arm_source>
ARM[:A][:LAYer[1]]:SOURce?
```

Parameters

<arm_source>	Query response
BUS	BUS
ECLTrg0	ECLT0
ECLTrg1	ECLT1
EXTernal	EXT
IECLTrg0	IECLTrg0
IECLTrg1	IECLTrg1
ITTLTrg0	ITTLTrg0
ITTLTrg1	ITTLTrg1
.	.
.	.
.	.
ITTLTrg7	ITTLTrg7
IMMediate (No source needed)	IMM
TTLTrg0	TTLT0
TTLTrg1	TTLT1
.	.
.	.
.	.
TTLTrg7	TTLT7

Reset Value

IMM

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set arm source to INTernal or any other invalid value.

Execution Error –212, “Arm ignored”
Sent *TRG or the VXIbus word serial <Trigger> command when ARM:SOURce is not set to BUS or when the instrument is not waiting at the ARM Event Detection layer.

Execution Error –215, “Arm deadlock”
Attempted to query the instrument when arm source is set to BUS and before sending *TRG or the VXIbus word serial <Trigger> command.

Dependencies

None

Examples

Command: `ARM:SOUR TTLTRG0`

Query: `ARM:SOUR?`

Response: `TTLT0`

Related Commands

TRIGger:SOURce

AVERage Subsystem

This section describes the commands in the [SENSe:]AVERage subsystem. See Figure 2–5. These commands select and setup one of four acquisition modes: normal, average, envelope, and peak detect. (Peak detect is available for TVS600A models only.) All active channels are affected by the acquisition mode selected. Averaging, peak detecting, and enveloping occur in the acquisition system before waveform records are passed to the CALC blocks. See Figure 2–6.

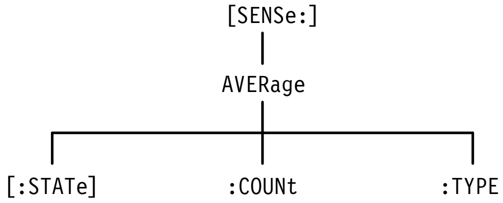


Figure 2–5: AVERage subsystem hierarchy

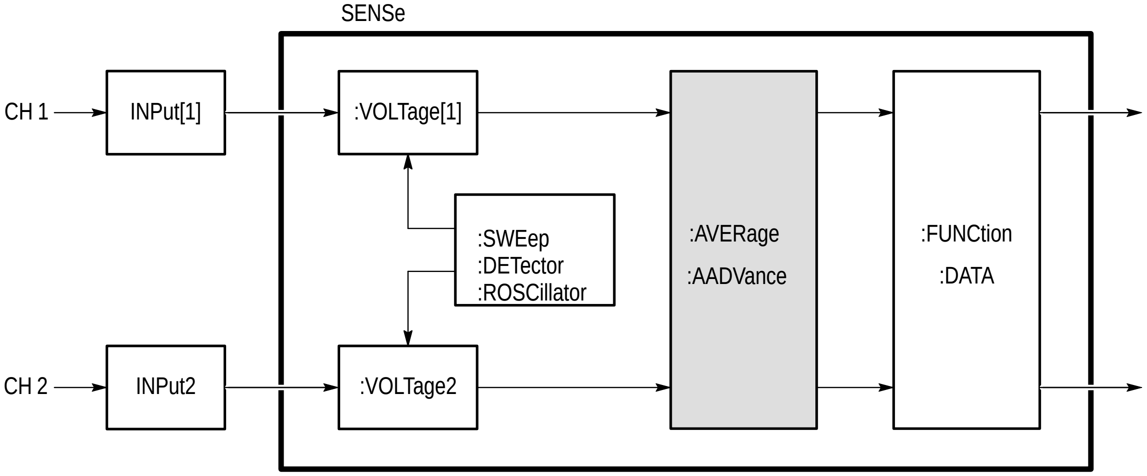


Figure 2–6: AVERage subsystem functional model

AVERage
AVERage?

Sets or queries whether the waveform analyzer performs averaging during acquisition. When the setting is off, the aquisition mode is normal; when on, the acquisition mode is either average, envelope or peak detect, depending on the setting for AVERage:TYPE (see page 2–16). The AVERage setting affects all channels. When averaging is on, the values returned from [SENSe:]DATA? are averaged, enveloped or peak-detected. The raw, unaveraged data is not available.

Syntax

```
[SENSe:]AVERage[:STATe] <boolean>
[SENSe:]AVERage[:STATe]?
```

Parameters

<boolean>	Query response
<NRf>	<NF1>
N ≠ 0 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

None

Dependencies

Enabling averaging sets AADVance to OFF.

Examples

Command: `AVER ON`

Query: `AVER?`

Response: 1

Related Commands

AVERage:COUNt
AVERage:TYPE
AADVance

AVERage:COUNt
AVERage:COUNt?

Sets or queries the number of acquisition records to average. Averaging reduces signal noise by approximately 3 dB for each power of 2 increase (2, 4, 8, 16) in the value of COUNt. For example, a COUNt setting of 8 will result in 3 dB less noise than a COUNt of 4. The COUNt setting affects all active channels. The count setting of one is intended for the PEAK DET type of averaging. PEAK DET uses hardware to envelope the acquired data. The other modes will function with a count of one, but with no added value.

Syntax

```
[SENSe:]AVERage:COUNt <count>
[SENSe:]AVERage:COUNt?
```

Parameters

<count>	Query response
<NRf> 1 ≤ N ≤ 4096	<NR1>
MINimum	2, 1[1]
MAXimum	4096

[1] **TVS600A only**

Reset Value

2

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set count to an illegal value.

Dependencies

None

Examples

Command: `AVER:COUN 16`

Query: `AVER:COUN?`

Response: `16`

Related Commands

AVERage
AVERage:TYPE

AVERage:TYPE
AVERage:TYPE?

Sets or queries the mode of acquisition to perform:

- SCALar (average), the default, averages each new sample point with the corresponding point in the previous acquisition record.
- ENVelope generates a waveform record of interleaved MAX/MIN pairs of sample points (the number of pairs is SWEep:POINts divided by two).
- PEAKdetect (TVS600A only) generates an array of MAX/MIN pairs, but uses hardware peak detect to capture glitches between sampling intervals.

The AVERage:TYPE setting affects all channels.

Peak Detect hardware is not functional at a sample rate faster than 100 MS/s. When peak detect is enabled on rates faster than 100 MS/s, the instrument will function as if in ENVelope mode and will not issue any warning or error to that effect.

Syntax

```
[SENSe:]AVERage:TYPE <type>
[SENSe:]AVERage:TYPE?
```

Parameters

<type>	Query response
ENVelope SCALar PEAKdetect[1]	ENV SCAL PEAK[1]

[1] **TVS600A only**

Reset Value

SCAL

Errors and Events

Command Error –141, ”Invalid character data”
Attempted to set average type to an illegal value.

Dependencies

None

Examples

Command: `AVER:TYPE ENV`

Query: `AVER:TYPE?`

Response: `ENV`

Related Commands

AVERage
AVERage:COUNt

CALCulate Subsystem

This section describes the commands in the CALCulate subsystems which process and perform measurements on acquisition records. These calculations are typically performed immediately after the waveform analyzer completes acquisition of the source. You can use CALCulate:IMMediate to reprocess and measure an existing acquisition record. Figures 2–7 and 2–8 show the command subblocks in the CALCulate subsystem.

For a description of how to process waveforms and make measurements with the commands in the CALCulate subsystem, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

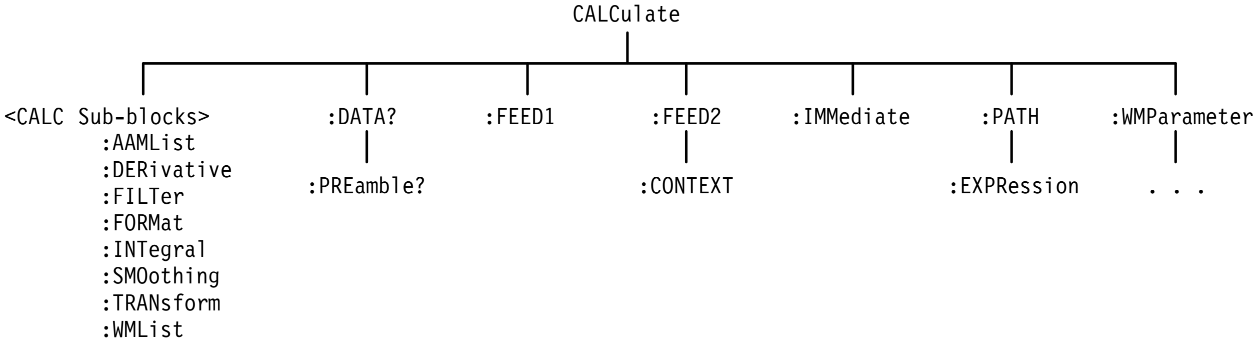


Figure 2–7: CALCulate subsystem hierarchy

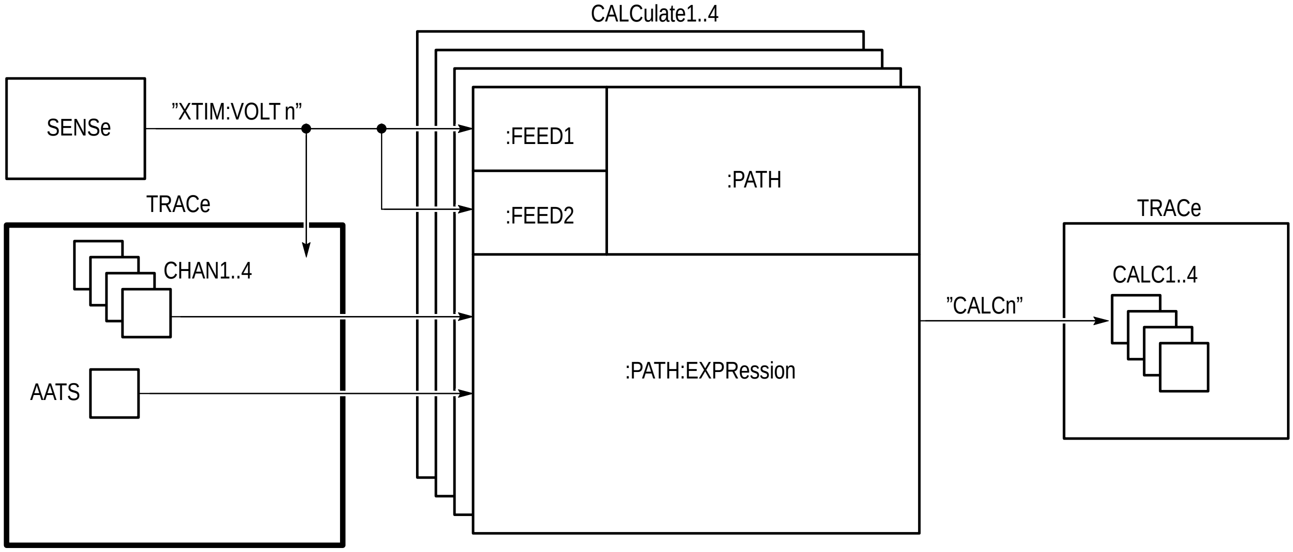


Figure 2–8: CALCulate subsystem functional model

CALCulate:AAMList
CALCulate:AAMList?

Sets or queries the list of measurements to perform on the selected auto-advance acquisition records. You can specify up to 50 measurements by separating them with commas. Before you can make auto-advance measurements, you must enable the auto-advance measurement system with the command CALCulate:AAMList:STATe.

Specify the measurement methods and reference values with the CALC:WMParameter commands.

The records on which measurements are made are specified by the AADV:REC:STARt and AADV:REC:COUNt commands.

Syntax

```
CALCulate<n>:AAMList <list>
CALCulate<n>:AAMList?
```

Parameters

<n>[1]	Query response
1 2 3 4	NA

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<list>	Query response
AMPLitude	AMPL
AREA	AREA
CARea	CAR
CMEan	CME
COPulse[1]	COP
CPARea[1]	CPAR
CRMS	CRMS
CROSs	CROSs
DELay	DEL
FREQuency	FREQ
FTIMe	FTIM
GAIN	GAIN
HIGH	HIGH
LOW	LOW
MAXimum	MAX
MEAN | DC	MEAN
MID	MID
MINimum	MIN
NCRoss	NCR
NDUTycycle	NDUT
NWIDth	NWID
OVERshoot[1]	OVER
PARea[1]	PAR
PCRoss	PCR
PERiod	PER
PDUTycycle	PDUT
PHAse	PHA
PREShoot[1]	PRES
PTPeak	PTP
PWIDth	PWID
RMS | AC	RMS
RTIMe	RTIM
SDEViation	SDEV
TTRig	TTR

[1] **TVS600A products only.**

Reset Value

MEAN

Errors and Events

Command Error 108, “Parameter not allowed”
Attempted to assign more than 50 measurements to the list.

Execution Error –141, “Invalid character data”
Attempted to program an illegal waveform measurement.

Dependencies

None

Examples

Command: `CALC1:AAML RTIM,FTIM,PWID`

Query: `CALC1:AAML?`

Response: `RTIM,FTIM,PWID`

Related Commands

CALCulate:AAMList:STATe
AADV:REC:STARt
AADV:REC:COUNt

CALCulate:AAMList:STATe CALCulate:AAMList:STATe?

Sets or queries whether to perform waveform measurement(s) on acquisition records captured with auto-advance acquisition. You specify Auto Advance measurements with the CALCulate:AAMList command.

Syntax

```
CALCulate<n>:AAMList:STATe <boolean>
CALCulate<n>:AAMList:STATe?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<boolean>	Query response
<NRf>	<NF1>
1 or ON	1
0 or OFF	0

Reset Value 0

Errors and Events None

Dependencies The setting for :STATe is ignored when you use the command CALC:PATH:EXPRession.

Examples

Command: `CALC:AAML:STAT ON`

Query: `CALC:AAML:STAT?`

Response: 1

Related Commands

CALCulate:AAMList
SWEep:ADDV ON|OFF

CALCulate:DATA? (Query Only)

This query returns the results of waveform processing and measurement functions performed on acquired waveform records. The default format of the returned data is ASCII. However, you can set the data format with the FORMat:CALCulate command.

Results are returned only when pending acquisitions and calculations are complete. The synchronizing commands *WAI, *OPC, and *OPC? are not required unless you wish to synchronize the transfer differently.

The CALCulate and TRACe subsystems use consistent naming, such that CALC1:DATA? is equivalent to TRAC? CALC1.

Syntax

```
CALCulate<n>:DATA?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

Reset Value

Not applicable

Errors and Events

Execution Error –230, “Data corrupt or stale”
Attempted to query data that is invalid, incomplete or stale.

Dependencies

None

Examples

Query: `CALC1:DATA?`

Response: `<arb_block_data>`

Related Commands

[SENSe:]DATA?
[SENSe:]DATA:PREamble?
CALCulate:DATA:PREamble?
TRACe?
TRACe:PREamble?

CALCulate:DATA:PREamble? (Query Only)

Returns the data preamble for acquisition record processing and measurement functions. Results are returned only when pending acquisitions and calculations are complete. The default format of the returned data is ASCII. However, you can set the data format with the FORMat:CALCulate command. The synchronizing commands *WAI, *OPC, and *OPC? are not required unless you wish to synchronize the transfer differently.

The CALCulate and TRACe subsystems use consistent naming, such that CALC1:DATA? is equivalent to TRAC? CALC1.

Syntax

```
CALCulate<n>:DATA:PREamble?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

Reset Value

Not applicable

Errors and Events

Execution Error –230, “Data corrupt or stale”
Attempted to query the preamble for data that is invalid, incomplete or stale.

Dependencies

None

Examples

Query: `CALC1:DATA:PRE?`

Response: `(DIF Expression)`

Related Commands

[SENSe:]DATA?
[SENSe:]DATA:PREamble?
CALCulate:DATA?
TRACe?
TRACe:PREamble?

CALCulate:DERivative:STATe CALCulate:DERivative:STATe?

Sets or queries whether to calculate a post-acquisition derivative on the selected channel. When ON, the waveform analyzer calculates the derivative for every point in an acquisition record. The result is a CALC<n> data record with the same number of points as the original acquisition record.

When you perform a calculation with the command CALC:PATH:EXPRession, the value of :STATe is ignored.

Syntax

```
CALCulate<n>:DERivative:STATe<boolean>
CALCulate<n>:DERivative:STATe?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<boolean> (DERivative state)	Query response
<NRf>	<NR1>
1 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command: `CALC1:DER:STAT ON`

Query: `CALC1:DER:STAT?`

Response: 1

Related Commands

CALCulate:INTegral:STATe

CALCulate:FEED[1]
CALCulate:FEED[1]?

Sets or queries the source of data for the specified CALCulate block. The calculations you set for a CALC block are performed on the source chosen with this command when using the CALC:PATH command. The source may be from any channel, reference, or a SENSe function; it is not tied to the CALC block number. For example, the FEED1 for CALC2 could be “XTIM:VOLT 1” or “XTIM:VOLT 3.” Additionally, more than one CALC block can operate on the same source.

Trace names and function strings may be used interchangeably for the source parameter <sense_func>. The query form always returns the function strings.

When you perform a calculation with the command CALC:PATH:EXPRession, the value of :FEED1 can be referenced as “%1” (or by name; CHAN1, for example) within the expression. Calculations are not performed on the :FEED1 source unless explicitly referenced in the CALC expression. See CALC:PATH and CALC:PATH:EXPR commands.

When specifying calculations that require more than one waveform, such as gain, phase, or delay, :FEED1 specifies the reference waveform and :FEED2 specifies the target waveform. The target waveform is measured with respect to the reference waveform; for example,1V in :FEED2 measured with respect to 2V in :FEED1 yields a gain (attenuation) of .5.

Syntax

```
CALCulate<n>:FEED[1] <sense_func>
CALCulate<n>:FEED[1]?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<sense_func> (CALC block source)			Query response
<string>			<string>
“XTIMe:VOLTage[:DC] 1”	or	CHAN1	“XTIM:VOLT 1”
“XTIMe:VOLTage[:DC] 2”	or	CHAN2	“XTIM:VOLT 2”
“XTIMe:VOLTage[:DC] 3”	or	CHAN3	“XTIM:VOLT 3”
“XTIMe:VOLTage[:DC] 4”	or	CHAN4	“XTIM:VOLT 4”
“REFerence1”	or	REF1	“REF1”
“REFerence2”	or	REF2	“REF2”
...			...
“REFerence10”	or	REF10	“REF10”
“ ”	or	NONE	“ ”

Reset Value

"" (None)

Errors and Events

Execution Error –141, “CALC1:FEED FTM”
Attempted to set FEED1 to an invalid trace name.

Execution Error –224, “Illegal parameter value, invalid sense function”
Attempted to set FEED1 to an invalid sense function.

Execution Error –241, “Hardware missing”
Attempted to set feed to “XTIM:VOLT 3” or “XTIM:VOLT 4” when the instrument is configured with two channels.

Dependencies

None

Examples

Command: `CALC1:FEED1 "XTIM:VOLT 1"`

Query: `CALC1:FEED1?`

Response: `"XTIM:VOLT 1"`

Related Commands

CALCulate:PATH

CALCulate:FEED2
CALCulate:FEED2?

Sets or queries a second source of data for the specified CALCulate block. The second source is used when measurements that are based on two waveforms, such as gain, phase, and delay, are specified using the CALCulate:WMList command.

The source may be from any channel, reference or a SENSe function, it is not tied to the CALC block number. For example, the FEED2 for CALC2 could be "XTIM:VOLT 1" or "XTIM:VOLT 3." Additionally, more than one CALC block can operate on the same source.

Trace names and function strings may be used interchangeably for the source parameter <sense_func>. The query form always returns the function strings.

When you perform a calculation with the command CALC:PATH:EXPRession, the value of :FEED2 can be referenced as "%2" within the expression.

When defining SCPI model calculations that measure parameters based on two waveforms (such as Gain and Delay), :FEED1 specifies the reference waveform and :FEED2 specifies the target waveform. Also, the functions you set using CALC:PATH only apply to the data source selected with FEED1. The source selected by FEED2 is unaffected by the functions you select.

Syntax

```
CALCulate<n>:FEED2 <sense_func>
CALCulate<n>:FEED2?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<sense_func> (CALC block source)			Query response
<string>			<string>
"XTIMe:VOLTage[:DC] 1"	or	CHAN1	"XTIM:VOLT 1"
"XTIMe:VOLTage[:DC] 2"	or	CHAN2	"XTIM:VOLT 2"
"XTIMe:VOLTage[:DC] 3"	or	CHAN3	"XTIM:VOLT 3"
"XTIMe:VOLTage[:DC] 4"	or	CHAN4	"XTIM:VOLT 4"
"REFerence1"	or	REF1	"REF1"
"REFerence2"	or	REF2	"REF2"
....			...
"REFerence10"	or	REF10	"REF10"
" "	or	NONE[1]	"" (empty string returned)

[1] **Specifying NONE causes :FEED2 to default to the same source as :FEED1. If you specify NONE, the same waveform will be used for taking the gain, phase, or delay measurement.**

Reset Value

`""` (NONE)

Errors and Events

Execution Error –141, "CALC1:FEED2 FTM"
Attempted to set FEED2 to an invalid trace name.

Execution Error –224, "Illegal parameter value, invalid sense function"
Attempted to set FEED2 to an invalid sense function.

Execution Error –241, "Hardware missing"
Attempted to set feed to "XTIM:VOLT 3" or "XTIM:VOLT 4" when the instrument is configured with two channels.

Dependencies

None

Examples

Command:	`CALC1:FEED2 "XTIM:VOLT 1"`
Query:	`CALC1:FEED2?`
Response:	`"XTIM:VOLT 1"`

Related Commands

CALCulate:PATH

CALCulate:FEED2:CONText
CALCulate:FEED2:CONText?

Sets or queries the measurement parameter block, one of CALC1–CALC4, used to characterize the FEED2 data.

Any of the four CALC blocks, CALC1–CALC4, may be used interchangeably for the context parameter <calc_block>.

Syntax

```
CALCulate<n>:FEED2:CONText <calc_block>
CALCulate<n>:FEED2:CONText?
```

Parameters

<calc_block> (CALC block source)	Query response
CALC1 CALC2 CALC3 CALC4	CALC1 CALC2 CALC3 CALC4
WMP1 WMP2 WMP3 WMP4	WMP1 WMP2 WMP3 WMP4

Reset Values

Context	Value
CALC1:FEED2:CONT	CALC1
CALC2:FEED2:CONT	CALC2
CALC3:FEED2:CONT	CALC3
CALC4:FEED2:CONT	CALC4

Errors and Events

Execution Error –141, “Invalid character data,
CALC1:FEED2:CONT FTM”
Attempted to set FEED2:CONT to an invalid parameter value.

Dependencies

None

Examples

Command: `CALC1:FEED2:CONT CALC4`

Query: `CALC1:FEED2:CONT?`

Response: `CALC4`

Related Commands

CALCulate:FEED1

CALCulate:FEED2

CALCulate:FILTer:FREQuency[:TYPE]
CALCulate:FILTer:FREQuency[:TYPE]?

Sets or queries the type of FREQuency filtering to perform on an acquisition record.

The available filter types are as follows:

- BPASs—rejects frequency components outside the defined frequency range.
- NOTCh—rejects frequency components within a defined frequency range.
- HPASs—rejects frequency components below a specified frequency.
- LPASs—rejects frequency components above a specified frequency.

When you set :TYPE to bandpass or notch, set the filter parameters with the :FILTer:FREQuency commands STARt, STOP, CENTer, and SPAN. When you set :TYPE to high pass or low pass, set the filter parameters with the commands HPASs or LPASs. For an overview of the measurement system, refer to the *TVS600 & TVS600A User Manual.*

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency[:TYPE] <type>
CALCulate<n>:FILTer[:GATE]:FREQuency[:TYPE]?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<type> (Frequency filter)	Query response
BPASs HPASs LPASs NOTCh	BPAS HPAS LPAS NOTC

Reset Value

BPAS

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set type to an illegal value.

Dependencies None

Examples

Command: `CALC1:FILT:FREQ NOTCH`

Query: `CALC1:FILT:FREQ?`

Response: `NOTC`

Related Commands

CALCulate:FILTer:FREQuency:CENTer
CALCulate:FILTer:FREQuency:HPASs
CALCulate:FILTer:FREQuency:LPASs
CALCulate:FILTer:FREQuency:SPAN
CALCulate:FILTer:FREQuency:STARt
CALCulate:FILTer:FREQuency:STATe
CALCulate:FILTer:FREQuency:STOP

CALCulate:FILTer:FREQuency:CENTer
CALCulate:FILTer:FREQuency:CENTer?

Sets or queries the center frequency of the bandpass or notch filter. Use the command :FREQuency:SPAN to set the frequency range for :NOTCh and BPASs frequency filters. For example, if you set :CENTer to 10 MHz and :SPAN to 2 MHz, then the :NOTCh filter will have a range of 9 MHz to 11 MHz. For an overview of the measurement system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:CENTer <center>
CALCulate<n>:FILTer[:GATE]:FREQuency:CENTer?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<center> (Center frequency of SPAN) [1]	Query response
<NRf>	<NR3>

[1] **The default multiplier for the <center> parameter is HZ for hertz. You can also use the multipliers KHZ for kilohertz, MHZ for megahertz, and GHZ for gigahertz.**

Reset Value

250.0E+6

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set center to an illegal value.

Dependencies

You can set the frequency range for the frequency filters :NOTCh and :BPASs with either the command pair :CENTer and :SPAN or the command pair :STARt and :STOP. Because both command pairs control the same parameters, changing either command pair will affect the other.

Examples

Command: `CALC1:FILT:FREQ:CENT 10E6`

Query: `CALC1:FILT:FREQ:CENT?`

Response: `10.0E+6`

Related Commands

CALCulate:FILTer:FREQuency:SPAN

CALCulate:FILTer:FREQuency:HPASs CALCulate:FILTer:FREQuency:HPASs?

Sets or queries the limit frequency (<cutoff>) below which the filter attenuates all frequency components. The :HPASs frequency is ignored when you select the bandpass, notch or low-pass filter. Use the command CALCulate:FILTer:FREQuency HPASs to enable the high-pass filter.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:HPASs <cutoff>
CALCulate<n>:FILTer[:GATE]:FREQuency:HPASs?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<cutoff> (Limit frequency)[1]	Query response
<NRf>	<NR3>

[1] **The default multiplier for the <cutoff> parameter is HZ for hertz. You can also use the multipliers KHZ for kilohertz, MHZ for megahertz, and GHZ for gigahertz.**

Reset Value

250.0E+6

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the high pass cutoff frequency to an illegal value.

Dependencies

None

Examples

Command: `CALC1:FILT:FREQ:HPAS 10E6`

Query: `CALC1:FILT:FREQ:HPAS?`

Response: `10.0E+6`

Related Commands

CALCulate:FILTer:FREQuency:LPASs

CALCulate:FILTer:FREQuency:LPASs
CALCulate:FILTer:FREQuency:LPASs?

Sets or queries the limit frequency (<cutoff>) above which the filter attenuates all frequency components. The :LPASs frequency is ignored when filter type is set to bandpass, notch or high pass. Use the command CALCulate:FILTer:FREQuency LPASs to enable the low-pass filter.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:LPASs <cutoff>
CALCulate<n>:FILTer[:GATE]:FREQuency:LPASs?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<cutoff> (Limit frequency)[1]	Query response
<NRf>	<NR3>

[1] The default multiplier for the <cutoff> parameter is HZ for hertz. You can also use the multipliers KHZ for kilohertz, MHZ for megahertz, and GHZ for gigahertz

Reset Value

250.0E+6

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the low-pass cutoff frequency to an illegal value.

Dependencies

None

Examples

Command: `CALC1:FILT:FREQ:LPAS 10E6`

Query: `CALC1:FILT:FREQ:LPAS?`

Response: `10.0E+6`

Related Commands

CALCulate:FILTer:FREQuency:HPASs

CALCulate:FILTer:FREQuency:SPAN
CALCulate:FILTer:FREQuency:SPAN?

Sets or queries the frequency range to be used by the bandpass and notch filters. After you define the range with :SPAN, you position the range within the available spectrum using the command :FREQuency:CENTer. For example, if you set :SPAN to 2 MHz and :CENTer to 10 MHz, then the :NOTCh filter will have a range of 9 MHz to 11 MHz.

The frequency span is ignored when you select the high-pass (:HPASs) or low-pass (:LPASs) filter.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:SPAN <span>
CALCulate<n>:FILTer[:GATE]:FREQuency:SPAN?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<span> (Frequency range)[1]	Query response
<NRf>	<NR3>

[1] **The default multiplier for the parameter <span> is HZ for hertz. You can also use the multipliers KHZ for kilohertz, MHZ for megahertz, and GHZ for gigahertz.**

Reset Value

100.0E+6

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set span to an illegal value.

Dependencies

You can set the frequency range for the frequency filters :NOTCh and :BPASs with either the command pair :CENTer and :SPAN or the command pair :STARt and :STOP. Because both command pairs control the same parameters, changing either command pair will affect the other.

Examples

Command: `CALC1:FILT:FREQ:SPAN 10E3`

Query: `CALC1:FILT:FREQ:SPAN?`

Response: `10.0E+3`

Related Commands

CALCulate:FILTer:FREQuency:CENTer

CALCulate:FILTer:FREQuency:SREJection CALCulate:FILTer:FREQuency:SREJection?

Sets or queries the level of rejection or attenuation for frequency components in the defined stop band for the:FREQuency filters. :SREJection sets the level of attenuation in dB for the filters :NOTCh, :BPASs, :LPASs and :HPASs. You can use :FREQuency:TWIDth to set the slope or roll off of the filter.

If you enter too large a value for :SREJection, the function will generate an execution error and terminate the filter process. If an execution error occurs, increase the record length or increase the value of :TWIDth. For an overview discussion of the waveform analyzer digital filter, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:SREJection <level>
CALCulate<n>:FILTer[:GATE]:FREQuency:SREJection?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<level> (Frequency rejection in dB)	Query response
<NRf>	<NR2>
15 ≤ N ≤ 100	
MINimum	15.0
MAXimum	100.0

Reset Value

60.0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set rejection to an illegal value.

Dependencies

None

Examples

Command: `CALC1:FILT:FREQ:SREJ 75`

Query: `CALC1:FILT:FREQ:SREJ?`

Response: `75.0`

Related Commands

CALCulate:FILTer:FREQuency:TWIDth

CALCulate:FILTer:FREQuency:STARt
CALCulate:FILTer:FREQuency:STARt?

Sets or queries the start, or lower limit, frequency of the bandpass and notch filters. :STARt and :STOP define the frequency range for the filters :FREQuency:BPASs and :FREQuency:NOTCh. The commands :FREQuency:CENTer and :FREQuency:SPAN provide an alternate way to set the filter range. For an overview discussion of the waveform analyzer digital filter, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

The :STARt frequency is ignored when you select the high-pass or low-pass filter.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:STARt <start>
CALCulate<n>:FILTer[:GATE]:FREQuency:STARt?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<start> (Beginning frequency of range)[1]	Query response
<NRf>	<NR3>

[1] The default multiplier for the parameter <start> is HZ for hertz. You can also use the multipliers KHZ for kilohertz, MHZ for megahertz, and GHZ for gigahertz.

Reset Value

200.0E+6

Errors and Events

Execution Error –222, “Data out of range”
Attempting to set start to an illegal value will generate execution error.

Dependencies

You can set the frequency range for the frequency filters :NOTCh and :BPASs with either the command pair :CENTer and :SPAN or the command pair :STARt and :STOP. Because both command pairs control the same parameters, changing either command pair will affect the other.

Examples

Command: `CALC1:FILT:FREQ:STAR 9.99E6`

Query: `CALC1:FILT:FREQ:STAR?`

Response: `9.99E+6`

Related Commands

CALCulate:FILTer:FREQuency:STOP
CALCulate:FILTer:FREQuency:CENTer
CALCulate:FILTer:FREQuency:SPAN

CALCulate:FILTer:FREQuency:STATe
CALCulate:FILTer:FREQuency:STATe?

Sets or queries whether the selected CALCulate block performs frequency filtering on acquisition records.

If you define an expression with CALC:PATH:EXPRession, then the value of STATe setting is ignored.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:STATe <boolean>
CALCulate<n>:FILTer[:GATE]:FREQuency:STATe?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<boolean> (Set filtering on or off)	Query response
<NRf>	<NR1>
1 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command: `CALC1:FILT:FREQ:STAT ON`

Query: `CALC1:FILT:FREQ:STAT?`

Response: 1

Related Commands

CALCulate:FILTer:FREQuency[:TYPE]

CALCulate:FILTer:FREQuency:STOP
CALCulate:FILTer:FREQuency:STOP?

Sets or queries the stop, or upper limit, frequency of the bandpass and notch filters. :STARt and :STOP define the frequency range for the filters :FREQuency:BPASs and :FREQuency:NOTCh. The commands :FREQuency:CENTer and :FREQuency:SPAN provide an alternate way to set the filter range.

The STOP frequency is ignored when filter type is set to high pass or low pass.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:STOP <stop>
CALCulate<n>:FILTer[:GATE]:FREQuency:STOP?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<stop> (Ending frequency of range)[1]	Query response
<NRf>	<NR3>

[1] **The default multiplier for the parameter <stop> is HZ for hertz. You can also use the multipliers KHZ for kilohertz, MHZ for megahertz, and GHZ for gigahertz.**

Reset Value

300.0E+6

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set stop to an illegal value.

Dependencies

You can set the frequency range for the frequency filters :NOTCh and :BPASs with either the command pair :CENTer and :SPAN or the command pair :STARt and :STOP. Because both command pairs control the same parameters, changing either command pair will affect the other.

Examples

Command: `CALC1:FILT:FREQ:STOP 10.01E6`

Query: `CALC1:FILT:FREQ:STOP?`

Response: `10.01E+6`

Related Commands CALCulate:FILTer:FREQuency:STARt
CALCulate:FILTer:FREQuency:CENTer
CALCulate:FILTer:FREQuency:SPAN

CALCulate:FILTer:FREQuency:TWIDth CALCulate:FILTer:FREQuency:TWIDth?

Sets or queries the slope of roll off for the post-acquisition filter. The filter slope is specified as a ratio of the absolute transition width (in Hz) to Nyquist frequency defined as 1 / (2 × <sample interval>). A low value for transition width produces a steep slope for the filter while a large value produces a gradual slope or roll off. You can also specify :TWIDth in percent.

If you enter too small a value for :TWIDth, the function will generate too many filter coefficients resulting in an execution error and termination of the filter process. For more information, refer to the filter discussion in the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
CALCulate<n>:FILTer[:GATE]:FREQuency:TWIDth <width>
CALCulate<n>:FILTer[:GATE]:FREQuency:TWIDth?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<width> (Rate of transition in percent)	Query response
<NRf>	<NR2>
0.0004 ≤ N ≤ 1.0	
MINimum	0.0004
MAXimum	1.0

Reset Value 0.1

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set transition width to an illegal value.

Dependencies

None

Examples

Command: `CALC1:FILT:FREQ:TWID 0.2`

Query: `CALC1:FILT:FREQ:TWID?`

Response: `0.2`

Related Commands

CALCulate:FILTer:FREQuency:SREJection

CALCulate:FORMat
CALCulate:FORMat?

Sets or queries whether to process the acquisition record to produce a new format from complex transform records. The available format types are:

- COMPlex—produce an xy pair for each sample point.
- MLINear—calculate the square root of the sum of the squares of x and y.
- MLOGarithmic—calculate 20*log$_{10}$ of the square root of the sum of the squares of x and y.
- NONE—produces no change in the acquisition record data.
- PHASe—calculates the arctan of y over x.
- POLar—calculates a pair of points r1 and r2, where r1 is the square root of the sum of the squares of x and y and r2 is the arctan of y over x.

Setting format to NONE effectively disables the function :FORMat.

When the source waveform is not complex and format is set to something other than NONE, the value for y in x +jy is assumed to be zero.

Syntax

```
CALCulate<n>:FORMat <format>
CALCulate<n>:FORMat?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<format>	Query response
COMPlex	COMP
MLINear	MLIN
MLOGarithmic	MLOG
NONE	NONE
PHASe	PHAS
POLar	POL

Reset Value

None

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set format to an illegal value.

Dependencies

None

Examples

Command: `CALC1:FORM MLOG`

Query: `CALC1:FORM?`

Response: `MLOG`

Related Commands

CALCulate:TRANsform:FREQuency

CALCulate:IMMediate
CALCulate:IMMediate?

Sets or queries whether the specified CALCulate block will reprocess SENSe data without reacquiring new data. The calculation is performed immediately and, if the query form is sent, the resulting data is returned afterward. The query form CALC:IMM? is semantically equivalent to CALC:IMM;DATA?. The format of the resulting data is determined by the FORMat subsystem.

This command sets the ACQ_OPC pending flag. *WAI, *OPC, or *OPC? may be used to synchronize the command.

Syntax

```
CALCulate<n>:IMMediate
CALCulate<n>:IMMediate?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<data>	Query response
Not applicable	Defined by FORMat:CALCulate

Reset Value

Not applicable

Errors and Events

The only errors are those associated with the defined measurements or calculations. Most calculation errors appear only at execution time rather than when you define a measurement list. The most common errors are Execution Error –260, “Expression error” and the Questionable Event errors: 2090, “Calculate1 questionable”; 2100, “Calculate2 questionable”; 2110, “Calculate3 questionable”; 2120, “Calculate4 questionable”.

Dependencies

None

Examples

Command: `CALC1:IMM`

Query: `CALC1:IMM?`

Response: `<data>`

Related Commands CALCulate:DATA?

CALCulate:INTegral:STATe
CALCulate:INTegral:STATe?

Sets or queries whether to process the acquisition record to produce an integral acquisition record.

The value of STATe is ignored when CALC:PATH:EXPRession is defined.

Syntax

```
CALCulate<n>:INTegral:STATe <boolean>
CALCulate<n>:INTegral:STATe?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<boolean> (Integral processing on or off)	Query response
<NRf>	<NR1>
1 or ON	1
0 or OFF	0

Reset Value 0

Errors and Events None

Dependencies None

Examples

Command: `CALC1:INT:STAT ON`

Query: `CALC1:INT:STAT?`

Response: 1

Related Commands CALCulate:DERivative:STATe

CALCulate:PATH CALCulate:PATH?

Sets or queries a list of CALCulate functions to execute in the order listed upon completion of acquisition. You specify the :PATH as a list of functions separated by commas. The CALCulate subsystem performs the calculations in sequential order on the source you define with CALCulate:FEED. CALCulate:PATH supports simple post-processing measurements. When a measurement cannot be expressed in a simple linear fashion, use the command CALC:PATH:EXPRession.

Calculations based on the CALC:PATH command are based on the SCPI calculation model. See the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a description of the SCPI model.

Syntax

```
CALCulate<n>:PATH <path>
CALCulate<n>:PATH?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<path> (Calculate subblocks to execute)	Query response
AAMList	AAML
DERivative	DER
FILTer	FILT
FORMat	FORM
INTegral	INT
SMOothing	SMO
TRANsform[1]	TRAN
WMList	WML

[1] TRANsform refers to the CALCulate:TRANsform:FREQuency commands.

Reset Value AAML, SMO, DER, INT, FILT, TRAN, FORM, WML

Errors and Events

Command Error 108, “Parameter not allowed”
Attempted to assign more than 12 functions to PATH.

Execution Error –141, “Invalid character data”
Attempted to program an invalid function.

Other calculation errors cannot be detected until execution time.

Dependencies

None

Examples

Command: `CALC1:PATH FILT,WML`

Query: `CALC1:PATH?`

Response: `FILT,WML`

Related Commands

CALCulate:AAMList
CALCulate:DERivative
CALCulate:FILTer
CALCulate:FORMat
CALCulate:INTegral
CALCulate:SMOothing
CALCulate:TRANsform:FREQuency
CALCulate:WMList

CALCulate:PATH:EXPRession
CALCulate:PATH:EXPRession?

Sets or queries a measurement expression using a “C like” programming language. When you define an :EXPRession it will be calculated in place of any calculation defined with the CALCulate:PATH command. The :EXPRession command allows you to perform calculations that cannot be expressed in a linear or sequential form.

Commands that control individual subblock operation also apply when the subblock is called by the CALC:PATH:EXPRession. For instance, whether you use CALC:PATH or CALC:PATH:EXPRession to call the high pass filter (CALC:FILT:FREQ:HPAS), you must still use CALC:FILT:FREQ:HPASs <cutoff> to set the lower limit frequency.

Sources to :EXPRession must be specified as trace names, such as CHAN1, instead of as data aliases, such as “XTIM:VOLT 1”. The settings of CALCulate:PATH and the :STATe commands of the subblocks are ignored. FEED1 and FEED2 can be accessed as %1, %2 respectively. For more information, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Calculations based on the CALC:PATH:EXPression command are based on the SCPI Expression model. See the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a description of the SCPI model.

Syntax

```
CALCulate<n>:PATH:EXPRession <path_expression>
CALCulate<n>:PATH:EXPRession?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<path_expression> (algebraic syntax expression)	Query response
<expression> (algebraic syntax expression)	<expression> (algebraic syntax expression)

Reset Value

( )

Errors and Events

Execution Error –241, "Hardware missing"
Attempted to define a path expression that requires CHAN3 or CHAN4 when the instrument is configured with two channels.

The :CALCulate subblock cannot detect most errors until execution time. If errors do occur, they are described with the failed :CALCulate function or with the description of any other failed system function.

Dependencies

None

Examples

Command: `CALC1:PATH:EXPR (RTIM(CHAN1+CHAN2))`

Query: `CALC1:PATH:EXPR?`

Response: `(RTIM(CHAN1+CHAN2))`

Related Commands

CALCulate:PATH

CALCulate:SMOothing CALCulate:SMOothing?

Sets or queries whether to perform smoothing on an acquisition record. Smoothing replaces each data point with the average for a specified number of adjacent data points. To set the number of data points to average, use :SMOothing:POINTs.

Syntax

```
CALCulate<n>:SMOothing[:STATe]<boolean>
CALCulate<n>:SMOothing[:STATe]?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<boolean> (Set smoothing on or off)	Query response
<NRf>	<NR1>
1 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command:	`CALC1:SMO ON`
Query:	`CALC1:SMO?`
Response:	1

Related Commands

CALCulate:SMOothing:POINts

CALCulate:SMOothing:POINts
CALCulate:SMOothing:POINts?

Sets or queries the number of adjacent points to average in the acquisition record when smoothing is enabled. Use CALCulate:SMOothing to enable smoothing for a particular CALCulate block.

The maximum number of data points to average is limited to one fourth the length of the acquisition record.

Syntax

```
CALCulate<n>:SMOothing:POINts <points>
CALCulate<n>:SMOothing:POINts?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<points>	Query response
<NRf> 2 ≤ N ≤ 65536	<NR1>
MINimum	2
MAXimum	<record length>

Reset Value 2

Errors and Events Execution Error –222, “Data out of range”
Attempted to set points to an illegal value.

Dependencies None

Examples

Command: `CALC1:SMO:POIN 8`

Query: `CALC1:SMO:POIN?`

Response: 8

Related Commands CALCulate:SMOothing

CALCulate:TRANsform:FREQuency:STATe
CALCulate:TRANsform:FREQuency:STATe?

Sets or queries whether to perform a Fast Fourier Transform (FFT) on the specified acquisition record to produce an equivalent frequency representation. The FFT function produces a complex array of pairs of real and imaginary data points. The number of data pairs in the resultant record is half the number of data points in the original record.

You can convert the complex data produced by the FFT function into magnitude or phase data using the CALC:FORMat block. Use the command :TRANsform:FREQuency:WINDow to select the type of data windowing (or shaping) used prior to the transformation.

Syntax

```
CALCulate<n>:TRANsform:FREQuency:STATe<boolean>
CALCulate<n>:TRANsform:FREQuency:STATe?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<boolean> (Set FFT transform on or off)	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value 0

Errors and Events None

Dependencies None

Examples Command: `CALC1:TRAN:FREQ:STAT ON`

Query: `CALC1:TRAN:FREQ:STAT?`

Response: 1

Related Commands

CALCulate:TRANsform:FREQuency:WINDow
CALCulate:FORMat

CALCulate:TRANsform:FREQuency:WINDow CALCulate:TRANsform:FREQuency:WINDow?

Sets or queries the type of data windowing (or shaping) to use prior to the FFT transformation. The FFT window acts as a bandpass filter. The window types and their typical use follow:

- BHARris — Widest pass band and lowest side lobes. Best for viewing a broad spectrum.
- BLACkman — Best window for measuring the amplitude of frequencies but worst at resolving frequencies.
- HAMMing — Very good window for resolving frequencies that are very close to the same value with somewhat improved amplitude accuracy over the rectangular window.
- HANNing — Very good window for measuring amplitude accuracy but degraded for resolving frequencies.
- RECTangular — Best type of window for resolving frequencies that are very close to the same value but worst for the accuracy of amplitude for those frequencies. Best type for measuring the frequency spectrum of nonrepetitive signals and measuring frequency components near DC.
- TRIangular — Least attenuation of side lobes. The triangular window is the convolution of two rectangles half the width of the window.

For more information on the FFT function and the use of window filters, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
CALCulate<n>:TRANsform:FREQuency:WINDow <window>
CALCulate<n>:TRANsform:FREQuency:WINDow?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<window>	Query response
BHARris	BHAR
BLACkman	BLAC
HAMMing	HAMM
HANNing	HANN
RECTangular	RECT
TRIangular	TRI

Reset Value

BHAR

Errors and Events

Execution Error –141, "Invalid character data"
Attempted to set window to an illegal value.

Dependencies

None

Examples

Command: `CALC1:TRAN:FREQ:WIND HAMM`

Query: `CALC1:TRAN:FREQ:WIND?`

Response: `HAMM`

Related Commands

CALCulate:TRANsform:FREQuency:STATe

CALCulate:WMList
CALCulate:WMList?

Sets or queries the list of waveform measurements to perform. Table 2–1 lists and describes the available measurements. All selected measurements are performed on the source chosen for the selected CALCulate block. Use the command CALCulate:WMList:STATe to enable the measurements. Refer to Appendix C for the measurement algorithms.

Measurement methods and reference values are specified by CALC:WMParameter commands. Each CALC block has a separate set of parameters.

Table 2–1: Waveform Measurement Definitions

Measurement	Definition
AMPLitude	Amplitude — the difference between HIGH and LOW. Current vertical units.
AREA	Area — the area across the entire acquisition record. Area above ground is positive and area below ground is negative. Current vertical units-seconds.
CARea	Cycle Area — the area across the first cycle of the acquisition record. Area above ground is positive; area below ground is negative. Current vertical units-seconds.
CMEan	Cycle Mean — the arithmetic mean of the first cycle in the acquisition record. Current vertical units.
COPulse[1]	Center Of Pulse — the arithmetic mean time of the six LREFerence, MREFerence, and HREFerence values for both the rising and falling edge of a pulse. Units are seconds.
CPARea[1]	Positive Area of Cycle — the arithmetic area of the absolute value of the waveform over the first cycle of the waveform. Current vertical units-seconds.
CRMS	Cycle RMS — the root-mean-square value for the deviation from ground of each point in the first cycle of the acquisition record. Current vertical units.
CROSs	Time at Crossing — the time relative to the trigger point of the specified crossing. The crossing is specified according to its sequence in the waveform record as an Nth crossing, with positive values for N referenced from the start and negative values referenced from the end of the waveform record. Units are seconds.
DELay	Delay — the time between the MidRef crossing on the specified edge of a reference waveform and the MidRef crossing on the specified edge a target waveform. Units are seconds.
FREQuency	Frequency — the reciprocal of the period of the first cycle in the acquisition record. Measured in Hz.

Table 2–1: Waveform Measurement Definitions (cont.)

Measurement	Definition
FTIMe	Fall Time — the time between the HREFerence crossing and the LREFerence crossing of the first falling edge of the acquisition record. Units are seconds.
Gain	Gain — the ratio of the amplitude of the target waveform to the amplitude of the reference waveform. Reference waveforms with zero amplitude return an error message. Gain has no units.
HIGH	High — the value used as 100% for calculating the HREFerence, MREFerence, and LREFerence. Current vertical units.
LOW	Low — the value used as 0% for calculating the HREFerence, MREFerence, and LREFerence. Current vertical units.
MAXimum	Maximum — the most positive value in the acquisition record. Current vertical units.
MEAN | DC	Mean or DC — the arithmetic mean of the entire acquisition record. Current vertical units.
MID	Middle — the mid-point between MINimum and MAXimum values. Current vertical units.
MINimum	Minimum — the most negative value in the acquisition record. Current vertical units.
NCROSs	Time at Nth Negative Crossing — the time relative to the trigger point of the specified negative crossing. The crossing is specified according to its sequence in the waveform record as an Nth negative crossing, with positive values for N referenced from the start and negative values referenced from the end of the waveform record. Positive crossings are ignored. Units are seconds.
NDUTycycle	Negative Dutycycle — the ratio between NWIDth and the PERiod of the acquisition record. No units.
NWIDth	Negative Width — the width of the first negative pulse in the acquisition record. Measured in seconds.
OVERshoot[1]	Overshoot of a pulse edge — the difference between the HIGH signal level (steady state response) and the positive peak amplitude (transient) Expressed as a percentage of the waveform amplitude. Contrast with PREshoot.
PARea[1]	Positive Area of Waveform — the arithmetic area of the absolute value of the waveform over the full measurement zone. Current vertical units-seconds.
PCROSs	Time at Nth Positive Crossing — the time relative to the trigger point of the specified positive crossing. Specified according to its sequence in the waveform record as an Nth positive crossing, with positive values for N referenced from the start and negative values referenced from the end of the waveform record. Negative crossings are ignored. Units are seconds.

Table 2–1: Waveform Measurement Definitions (cont.)

Measurement	Definition
PDUTycycle	Positive Dutycycle — the ratio between PWIDth and the PERiod of the acquisition record. No units.
PERiod	Period — the width of the first cycle in the acquisition record. Measured in seconds.
Phase	Phase — the lead or lag in degrees between the MidRef crossing on the specified edge of a target waveform and the MidRef crossing on the specified edge a reference waveform. Phase is positive when target leads reference waveform; this measurement uses period of the target waveform when computing degrees of phase. Units are degrees.
PREShoot[1]	Preshoot of a pulse edge — the difference between the LOW signal level (steady state response) and the negative peak amplitude (transient) immediately preceding it. Expressed as a percentage of the waveform amplitude. Contrast with OVERshoot.
PTPeak	Peak To Peak — the difference between MAXimum and MINimum. Current vertical units.
PWIDth	Positive Width — the width of the first positive pulse in the acquisition record. Measured in seconds.
RMS | AC	RMS or AC — the root-mean-square value for the deviation from ground of each point in the complete acquisition record. Current vertical units.
RTIMe	Rise Time — the time between the LREFerence crossing and the HREFerence crossing of the first rising edge of the acquisition record. Measured in seconds.
SDEViation	Standard Deviation — the root-mean-square value for the deviation from the arithmetic mean of each point in the acquisition record. Current vertical units.
TTRig	Trigger-to-trigger time — the time between the trigger event in the main acquistion and the trigger event of the delayed acquisition. Measured in seconds.

[1] **TVS600A only**

Syntax

```
CALCulate<n>:WMList <list>
CALCulate<n>:WMList?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<list>	Query response
AMPLitude	AMPL
AREA	AREA
CARea	CAR
CMEan	CME
COPulse[1]	COP
CPARea[1]	CPAR
CRMS	CRMS
CROSs	CROSs
DELay	DEL
FREQuency	FREQ
FTIMe	FTIM
GAIN	GAIN
HIGH	HIGH
LOW	LOW
MAXimum	MAX
MEAN | DC	MEAN
MID	MID
MINimum	MIN
NCRoss	NCR
NDUTycycle	NDUT
NWIDth	NWID
OVERshoot[1]	OVER
PARea[1]	PAR
PCRoss	PCR
PERiod	PER
PDUTycycle	PDUT
PHASe	PHAS
PREShoot[1]	PRES
PTPeak	PTP
PWIDth	PWID
RMS | AC	RMS
RTIMe	RTIM
SDEViation	SDEV

<list> (cont.)	Query response (cont.)
TTRig	TTR

[1] **TVS600A products only.**

Reset Value MEAN

Errors and Events Execution Error –141, “Invalid character data”
Attempted to program an illegal waveform measurement.

Dependencies None

Examples Command: `CALC1:WML RTIM,FTIM,PWID`

Query: `CALC1:WML?`

Response: `RTIM,FTIM,PWID`

Related Commands CALCulate:WMList:STATe

CALCulate:WMList:STATe
CALCulate:WMList:STATe?

Sets or queries whether the waveform measurement list for the specified CALC block will execute after the next acquisition.

Syntax

```
CALCulate<n>:WMList:STATe <boolean>
CALCulate<n>:WMList:STATe?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<boolean>	Query response
<NRf>	<NF1>
1 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

None

Dependencies

The setting for :STATe is ignored when you use the command CALC:PATH:EXPRession.

Examples

Command: `CALC1:WML:STAT ON`

Query: `CALC1:WML:STAT?`

Response: 1

Related Commands

CALCulate:WMList

CALCulate:WMParameter:EDGE CALCulate:WMParameter:EDGE?

Sets or queries the index of the edge desired for cross and delay waveform measurements. This setting affects FTIMe, RTIMe, CROSs, NCRoss, PCRoss DELay, and COPulse measurements. A positive <edge> value searches from the beginning of the waveform record. A negative <edge> value searches from the end of the waveform record. Zero <edge> is the last edge in the record. Measurement functions with implcitly defined slope (FTIMe, RTIMe, NCRoss, PCRoss) or explicitly defined slope (DELay) only count edges of the proper slope. CROSs and COPulse count edges of either slope.

Syntax

```
CALCulate[1]:WMParameter:EDGE <edge>
CALCulate[1]:WMParameter:EDGE?
.
.
CALCulate4:WMParameter:EDGE <edge>
CALCulate4:WMParameter:EDGE?
```

Parameters

<edge>	Query response
<NRf>	<NR1>

Suffix Units and Multipliers None (default vertical units)

Reset Value 1

Errors and Events Execution Error –222, “Data out of range”
Attempted to set EDGE to an illegal value.

Dependencies None

Examples

Command: `CALC1:WMP:EDGE 4`

Query: `CALC1:WMP:EDGE?`

Response: `4`

Related Commands CALCulate:WMParameter:LOW

CALCulate:WMParameter:GATE CALCulate:WMParameter:GATE?

TVS600A Models Only

Sets or queries the state of measurement gating. Settings established with other GATE parameters are ignored unless CALC:WMP:GATE is set to ON.

Syntax

```
CALCulate[1]:WMParameter:GATE <boolean>
CALCulate[1]:WMParameter:GATE?
.
.
CALCulate4:WMParameter:GATE <boolean>
CALCulate4:WMParameter:GATE?
```

Parameters

<boolean>	Query response
<NRf>	<NR1>
N ≠ 0	1
0	0
ON	1
OFF	0

Suffix Units and Multipliers None

Reset Value 0

Errors and Events None

Dependencies None

Examples

Command: `CALC1:WMP:GATE ON`

Query: `CALC1:WMP:GATE?`

Response: 1

Related Commands

CALCulate:WMParameter:GATE:MODe
CALCulate:WMParameter:GATE:STARt
CALCulate:WMParameter:GATE:STOP

CALCulate:WMParameter:GATE:METHod CALCulate:WMParameter:GATE:METHod?

TVS600A Models Only

Sets or queries the method of determining gate location for waveform measurements. This setting is ignored unless CALC:WMP:GATE is set to ON.

Syntax

```
CALCulate[1]:WMParameter:GATE:METHod <mode>
CALCulate[1]:WMParameter:GATE:METHod?
.
.
CALCulate4:WMParameter:GATE:METHod <mode>
CALCulate4:WMParameter:GATE:METHod?
```

Parameters

<mode>	Query response
ABSolute RELative	ABS REL

Suffix Units and Multipliers

None

Reset Value

REL

Errors and Events

None

Dependencies

None

Examples

Command: `CALC1:WMP:GATE:METH ABS`

Query: `CALC1:WMP:GATE:METH?`

Response: `ABS`

Related Commands

CALCulate:WMParameter:GATE
CALCulate:WMParameter:GATE:STARt
CALCulate:WMParameter:GATE:STOP

CALCulate:WMParameter:GATE:STARt:ABSolute CALCulate:WMParameter:GATE:STARt:ABSolute?

TVS600A Models Only

Sets or queries the absolute starting location for gated waveform measurements.

This setting is ignored unless CALC:WMP:GATE is set to ON and CALC:WMP:GATE:METH is set to ABS.

Syntax

```
CALCulate[1]:WMParameter:GATE:STARt[:ABSolute] <location>
CALCulate[1]:WMParameter:GATE:STARt[:ABSolute]?
.
.
CALCulate4:WMParameter:GATE:STARt[:ABSolute] <location>
CALCulate4:WMParameter:GATE:STARt[:ABSolute]?
```

Parameters

<location>	Query response
<NRf>	<NR3>

Suffix Units and Multipliers None

Reset Value –9.9e+37

Errors and Events None

Dependencies None

Examples

Command: `CALC1:WMP:GATE:STAR 10.0`

Query: `CALC1:WMP:GATE:STAR?`

Response: `10.0E+0`

Related Commands

CALCulate:WMParameter:GATE
CALCulate:WMParameter:GATE:METHod
CALCulate:WMParameter:GATE:STOP:ABSolute

CALCulate:WMParameter:GATE:STARt:RELative CALCulate:WMParameter:GATE:STARt:RELative?

TVS600A Models Only

Sets or queries the relative starting location for gated waveform measurements.

This setting is ignored unless CALC:WMP:GATE is set to ON and CALC:WMP:GATE:METH is set to REL.

Syntax

```
CALCulate[1]:WMParameter:GATE:STARt:RELative <ratio>
CALCulate[1]:WMParameter:GATE:STARt:RELative?
.
.
CALCulate4:WMParameter:GATE:STARt:RELative <ratio>
CALCulate4:WMParameter:GATE:STARt:RELative?
```

Parameters

<ratio>	Query response
<NRf>	<NR3>

Suffix Units and Multipliers

PCT

Reset Value

0.0

Errors and Events

None

Dependencies

None

Examples

Command: `CALC1:WMP:GATE:STAR:REL 0.0`

Query: `CALC1:WMP:GATE:STAR:REL?`

Response: `0.0E+0`

Related Commands

CALCulate:WMParameter:GATE
CALCulate:WMParameter:GATE:METHod
CALCulate:WMParameter:GATE:STOP:RELative

CALCulate:WMParameter:GATE:STOP:ABSolute CALCulate:WMParameter:GATE:STOP:ABSolute?

TVS600A Models Only

Sets or queries the absolute stopping location for gated waveform measurements.

This setting is ignored unless CALC:WMP:GATE is set to ON and CALC:WMP:GATE:METH is set to ABS.

Syntax

```
CALCulate[1]:WMParameter:GATE:STOP[:ABSolute] <location>
CALCulate[1]:WMParameter:GATE:STOP[:ABSolute]?
.
.
CALCulate4:WMParameter:GATE:STOP[:ABSolute] <location>
CALCulate4:WMParameter:GATE:STOP[:ABSolute]?
```

Parameters

<location>	Query response
<NRf>	<NR3>

Suffix Units and Multipliers None

Reset Value 9.9e+37

Errors and Events None

Dependencies None

Examples

Command: `CALC1:WMP:GATE:STOP 10.0`

Query: `CALC1:WMP:GATE:STOP?`

Response: `10.0E+0`

Related Commands

CALCulate:WMParameter:GATE
CALCulate:WMParameter:GATE:METHod
CALCulate:WMParameter:GATE:STARt:ABSolute

CALCulate:WMParameter:GATE:STOP:RELative CALCulate:WMParameter:GATE:STOP:RELative?

TVS600A Models Only

Sets or queries the relative stopping location for gated waveform measurements.

This setting is ignored unless CALC:WMP:GATE is set to ON and CALC:WMP:GATE:METH is set to REL.

Syntax

```
CALCulate[1]:WMParameter:GATE:STOP:RELative <ratio>
CALCulate[1]:WMParameter:GATE:STOP:RELative?
.
.
CALCulate4:WMParameter:GATE:STOP:RELative <ratio>
CALCulate4:WMParameter:GATE:STOP:RELative?
```

Parameters

<ratio>	Query response
<NRf>	<NR3>

Suffix Units and Multipliers

PCT

Reset Value

1.0

Errors and Events

None

Dependencies

None

Examples

Command: `CALC1:WMP:GATE:STOP:REL 1.0`

Query: `CALC1:WMP:GATE:STOP:REL?`

Response: `1.0E+0`

Related Commands

CALCulate:WMParameter:GATE
CALCulate:WMParameter:GATE:METHod
CALCulate:WMParameter:GATE:STARt:RELative

CALCulate:WMParameter:HIGH CALCulate:WMParameter:HIGH?

Sets or queries the high (most positive) level used for time and amplitude measurements. This setting is used only when CALC:WMP:HMEThod is set to ABSolute. For more information, refer to the discussion of CALC:WMP:HMEThod. The units are those currently used for vertical magnitude.

Syntax

```
CALCulate<n>:WMParameter:HIGH <high>
CALCulate<n>:WMParameter:HIGH?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<high> (Absolute HIGH level)	Query response
<NRf>	<NR3>

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set HIGH to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:HIGH 10.0`

Query: `CALC1:WMP:HIGH?`

Response: `10.0E+0`

Related Commands

CALCulate:WMParameter:HMEThod
CALCulate:WMParameter:LOW

CALCulate:WMParameter:HMEThod CALCulate:WMParameter:HMEThod?

Sets or queries the method for calculating the HIGH (most positive) value for time and amplitude waveform measurements.

ABSolute — specifies that HIGH is set to the value of CALCulate:WMParameter:HIGH.

AUTO — selects the MODE method of setting HIGH when the histogram function is able to detect a consistent level above MID. Otherwise, the PEAK method is used. This method is effective when you are not certain what type of waveform to expect.

MODE — selects the level for HIGH based on a peak histogram function which looks for a greater than average number of data points at a level above MID. This method is useful to ignore spurious peaks on a digital logic waveform, such as a TTL clock signal.

PEAK — specifies that HIGH is set to the highest amplitude data point in the acquisition record. This method is useful for a sinewave or triangle waveform.

Syntax

```
CALCulate<n>:WMParameter:HMEThod <method>
CALCulate<n>:WMParameter:HMEThod?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<method> (Select how to set HIGH)	Query response
ABSolute AUTO MODE PEAK	ABS AUTO MODE PEAK

Reset Value MODE

Errors and Events

Execution Error –141, “Invalid character data,
CALC1:WMP FTM”
Attempted to set method to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:HMET ABS`

Query: `CALC1:WMP:HMET?`

Response: `ABS`

Related Commands

CALCulate:WMParameter:LMEThod
CALCulate:WMParameter:HIGH

CALCulate:WMParameter:LOW
CALCulate:WMParameter:LOW?

Sets or queries the low (most negative) level used for time and amplitude measurements. This setting is used only when CALC:WMP:LMEThod is set to ABSolute. For more information, refer to the discussion of CALC:WMP:HMEThod. The units are those currently used for vertical magnitude.

Syntax

```
CALCulate<n>:WMParameter:LOW <low>
CALCulate<n>:WMParameter:LOW?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<low> (Absolute LOW level)	Query response
<NRf>	<NR3>

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set LOW to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:LOW -10.0`

Query: `CALC1:WMP:LOW?`

Response: `-10.0E+0`

Related Commands

CALCulate:WMParameter:HIGH
CALCulate:WMParameter:LMEThod

CALCulate:WMParameter:LMEThod
CALCulate:WMParameter:LMEThod?

Sets or queries the method for calculating the LOW (most negative) value for time and amplitude waveform measurements.

ABSolute — specifies that LOW is set to the value of CALCulate:WMParameter:LOW.

AUTO — selects the MODE method of setting LOW when the histogram function is able to detect a consistent level below MID. Otherwise, the PEAK method is used. The AUTO method is effective when you are not certain what type of waveform to expect.

MODE — selects the level for LOW based on a peak histogram function which looks for a greater than average number of data points at a level below MID. This method is useful to ignore spurious peaks on a digital logic waveform, such as a TTL clock signal.

PEAK — specifies that LOW is set to the lowest amplitude data point in the acquisition record. This method is useful for a sinewave or triangle waveform.

Syntax

```
CALCulate<n>:WMParameter:LMEThod <method>
CALCulate<n>:WMParameter:LMEThod?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<method> (Select how to set LOW)	Query response
ABSolute AUTO MODE PEAK	ABS AUTO MODE PEAK

Reset Value

MODE

Errors and Events

Execution Error –141, "Invalid character data"
Attempted to set method to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:LMET ABS`

Query: `CALC1:WMP:LMET?`

Response: `ABS`

Related Commands

CALCulate:WMParameter:HMEThod
CALCulate:WMParameter:LOW

CALCulate:WMParameter:HREFerence
CALCulate:WMParameter:HREFerence?

Sets or queries the high reference (distal) level in vertical units for time and amplitude measurements. This setting is used when you have set CALC:WMP:RMEThod to ABSolute.

Syntax

```
CALCulate<n>:WMParameter:HREFerence[:ABSolute] <href>
CALCulate<n>:WMParameter:HREFerence[:ABSolute]?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<href> (Set absolute value of HREF)	Query response
<NRf>	<NR3>

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the high reference to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:HREF 8`

Query: `CALC1:WMP:HREF?`

Response: `8.0E+0`

Related Commands

CALCulate:WMParameter:HREFerence:RELative
CALCulate:WMParameter:RMEThod

CALCulate:WMParameter:HREFerence:RELative CALCulate:WMParameter:HREFerence:RELative?

Sets or queries the high reference (distal) level used for time and amplitude measurements. The level can be expressed as a ratio or percent of the current value for CALC:WMP:AMPLitude. This setting is used when you have set CALC:WMP:RMEThod to RELative.

Syntax

```
CALCulate<n>:WMParameter:HREFerence:RELative <href>
CALCulate<n>:WMParameter:HREFerence:RELative?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<href> (Set relative value of HREF)	Query response
<NRf>	<NR2>
$0.0 \leq N \leq 1.0$[1]	
MINimum	0.0
MAXimum	1.0

[1] **The units must be specified as PCT when setting HREFerence:RELative as a percent of the value AMPL.**

Reset Value

0.9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the high reference to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:HREF:REL 95 PCT`

Query: `CALC1:WMP:HREF:REL?`

Response: `0.95`

Related Commands CALCulate:WMParameter:HREFerence

CALCulate:WMParameter:LREFerence CALCulate:WMParameter:LREFerence?

Sets or queries the low reference (proximal) level in vertical units for time and amplitude measurements. This setting is used only when CALC:WMP:RMEThod is set to ABSolute.

Syntax

```
CALCulate<n>:WMParameter:LREFerence[:ABSolute] <lref>
CALCulate<n>:WMParameter:LREFerence[:ABSolute]?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<lref> (Set absolute value of LREF)	Query response
<NRf>	<NR2>

Reset Value 0.0E+0

Errors and Events Execution Error –222, “Data out of range”
Attempted to set the low reference to an illegal value.

Dependencies None

Examples

Command: `CALC1:WMP:LREF –8`

Query: `CALC1:WMP:LREF?`

Response: `–8.0E+0`

Related Commands CALCulate:WMParameter:LREFerence:RELative
CALCulate:WMParameter:RMEThod

CALCulate:WMParameter:LREFerence:RELative CALCulate:WMParameter:LREFerence:RELative?

Sets or queries the low reference (proximal) level used for time and amplitude measurements. The level can be expressed as a ratio or percent of the current value for CALC:WMP:AMPLitude. This setting is used only when CALC:WMP:RMEThod is set to RELative.

Syntax

```
CALCulate<n>:WMParameter:LREFerence:RELative <lref>
CALCulate<n>:WMParameter:LREFerence:RELative?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<lref> (Set relative value of LREF)	Query response
<NRf>	<NR2>
0.0 ≤ N ≤ 1.0[1]	
MINimum	0.0
MAXimum	1.0

[1] **The units must be specified as PCT when setting LREFerence:RELative as a percent of the value AMPL.**

Reset Value

0.1

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the low reference to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:LREF:REL 5 PCT`

Query: `CALC1:WMP:LREF:REL?`

Response: `0.5`

Related Commands

CALCulate:WMParameter:LREFerence
CALCulate:WMParameter:RMEThod

CALCulate:WMParameter:MREFerence CALCulate:WMParameter:MREFerence?

Sets or queries the middle reference (mesial) level in vertical units for time and amplitude measurements. The MREFerence setting is effective only when CALC:WMP:RMEThod is set to ABSolute.

Syntax

```
CALCulate<n>:WMParameter:MREFerence[:ABSolute] <mref>
CALCulate<n>:WMParameter:MREFerence[:ABSolute]?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<mref> (Set absolute value of MREF)	Query response
<NRf>	<NR3>

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the middle reference to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:MREF 1`

Query: `CALC1:WMP:MREF?`

Response: `1.0E+0`

Related Commands CALCulate:WMParameter:MREFerence:RELative
CALCulate:WMParameter:RMETHod

CALCulate:WMParameter:MREFerence:HYSTeresis
CALCulate:WMParameter:MREFerence:HYSTeresis?

Sets or queries the middle reference (mesial) hysteresis used for time and amplitude measurements. The hysteresis can be expressed as a ratio or percent of the current value for AMPLitude. The signal must transition beyond the vertical range defined by :MREFerence:HYSTeresis before another crossing of :MREFerence can be accepted for measurements such as :PERiod.

This setting is used only when CALC:WMP:RMETHod is set to RELative.

Syntax

```
CALCulate<n>:WMParameter:MREFerence:HYSTeresis <hyst>
CALCulate<n>:WMParameter:MREFerence:HYSTeresis?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<hyst> (Set relative value of HYST)	Query response
<NRf>	<NR2>
0.0 ≤ N ≤ 0.5[1]	
MINimum	0.0
MAXimum	0.5

[1] The units must be specified as PCT when setting MREFerence:HYSTeresis as a percent of the value AMPL.

Reset Value 0.05

Errors and Events Execution Error –222, “Data out of range”
Attempted to set the middle reference hysteresis to an illegal value.

Dependencies None

Examples

Command: `CALC1:WMP:MREF:HYST 10 PCT`

Query: `CALC1:WMP:MREF:HYST?`

Response: `0.1`

Related Commands

CALCulate:WMParameter:MREFerence

CALCulate:WMParameter:MREFerence:RELative CALCulate:WMParameter:MREFerence:RELative?

Sets or queries the middle reference (mesial) level used for time and amplitude measurements. The level can be expressed as a ratio or percent of the current value for AMPLitude. This setting is used only when CALC:WMP:RMEThod is set to RELative.

Syntax

```
CALCulate<n>:WMParameter:MREFerence:RELative <mref>
CALCulate<n>:WMParameter:MREFerence:RELative?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] **The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.**

<mref> (Set relative value of MREF)	Query response
<NRf> 0.0 ≤ N ≤ 1.0[1]	<NR2>
MINimum	0.0
MAXimum	1.0

[1] **The units must be specified as PCT when setting MREFerence:RELative as a percent of the value AMPL.**

Reset Value

0.5

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the middle reference to an illegal value.

Dependencies None

Examples

Command: `CALC1:WMP:MREF:REL 55 PCT`

Query: `CALC1:WMP:MREF:REL?`

Response: `0.55`

Related Commands

CALCulate:WMParameter:MREFerence
CALCulate:WMParameter:RMEThod

CALCulate:WMParameter:RMEThod CALCulate:WMParameter:RMEThod?

Sets or queries the method for calculating the reference (high, middle, low) values for time and amplitude measurements. Selecting ABSolute lets you set the measurement parameters at absolute vertical levels. RELative lets you set them as a ratio or percentage of the parameter to the vertical amplitude of the record data.

Syntax

```
CALCulate<n>:WMParameter:RMEThod <method>
CALCulate<n>:WMParameter:RMEThod?
```

Parameters

<n>[1]	Query response
1 2 3 4	Not applicable

[1] The calculate block number <n> selects one of four calculate blocks. If you omit <n>, the default is calculate block 1.

<method> (Sets reference method)	Query response
ABSolute RELative	ABS REL

Reset Value

REL

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set method to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:RMET ABS`

Query: `CALC1:WMP:RMET?`

Response: `ABS`

Related Commands

CALCulate:WMParameter:HREFerence
CALCulate:WMParameter:LREFerence
CALCulate:WMParameter:MREFerence

CALCulate:WMParameter:SLOPe
CALCulate:WMParameter:SLOPe?

Sets or queries the slope of the edge desired for delay waveform measurements. This setting only affects DELay measurements. All other edge measurements determine the edge slope implicitly from the measurement specified. For example, RTIMe uses positive slope.

Syntax

```
CALCulate[1]:WMParameter:SLOPe <slope>
CALCulate[1]:WMParameter:SLOPe?
CALCulate[2]:WMParameter:SLOPe <slope>
CALCulate[2]:WMParameter:SLOPe?.
CALCulate[3]:WMParameter:SLOPe <slope>
CALCulate[3]:WMParameter:SLOPe?.
CALCulate[4]:WMParameter:SLOPe <slope>
CALCulate[4]:WMParameter:SLOPe?
```

Parameters

<slope>	Query response
NEGative	NEG
POSitive	POS

Suffix Units and Multipliers

None (default vertical units)

Reset Value

POS

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set SLOPe to an illegal value.

Dependencies

None

Examples

Command: `CALC1:WMP:SLOP NEG`

Query: `CALC1:WMP:SLOP?`

Response: `NEG`

Related Commands

CALCulate:WMParameter:EDGE

CALibration Subsystem

This section describes the commands in the CALibration subsystem. See Figure 2–9. These commands run the waveform analyzer self-calibration functions.

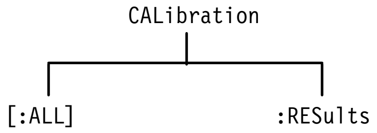


Figure 2–9: CALibration subsystem hierarchy

CALibration
CALibration?

Executes all self-calibration functions. If a failure occurs, calibration will immediately stop. The query form runs the self-calibration functions and returns the numeric identifier of the first failed function. A value of zero indicates there were no failures. The command form executes the functions but returns no results code.

Calibration functions have numeric identifiers in the range 2000 to 2999. CALibration sets the CAL_OPC pending flag. Use the command *WAI, *OPC, or *OPC? to synchronize the response with completion of self calibration.

To compensate probes, use the command OUTPut:PCOMpensate which enables the probe calibration signal to the front panel.

Syntax

```
CALibration[:ALL]
CALibration[:ALL]?
```

Parameters

<boolean>	Query response
Not applicable	<NR1> 0 No failures 2000 ≤ N ≤ 2999

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `CAL`

Query: `CAL?`

Response: `0`

Related Commands

CALibration:RESults?
CALibration:RESults:VERBose?

CALibration:PROBe
CALibration:PROBe?

TVS600A Models Only

This command calibrates the gain of the attached active voltage probes, but returns no results code. The query returns a value of zero if the probe is calibrated.

This command only works with active voltage probes.

Upon successful completion of the probe cal, the calibration constants are stored nonvolatile along with the probe identification and serial number. These cal constants are used as long as the identification and serial numbers match the attached probe.

1. A value of one (1) returned by the query indicates an initialized (uncalibrated) probe.
2. A value of zero (0) indicates a calibrated probe.
3. A value of minus one (–1) indicates a probe calibration failure.
4. A value of minus two (–2) indicates a probe calibration failed because the probe was not connected to the calibration source.
5. A value of minus three (–3) indicates a calibration failed because an unsupported probe or no probe was installed.

The current state of the instrument is saved before the cal and restored after.

This command/query sets the CAL_OPC pending flag. *WAI, *OPC, or *OPC? may be used to synchronize the command.

Syntax

```
CALibration:PROBe[1]
CALibration:PROBe[1]?
CALibration:PROBe2
CALibration:PROBe2?
CALibration:PROBe3
CALibration:PROBe3?
CALibration:PROBe4
CALibration:PROBe4?
```

Parameters

<calibration–status>	Query response
Not applicable	<NR1> 0 1 –1 –2 –3

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

Execution Error –240, “Hardware error”. A failed calibration occurred. CAL:PROB? will return –1.

Execution Error –241, “Hardware missing”. Attempted to calibrate a probe with no level 2 probe attached.

Execution Error –241, “Hardware missing”. Attempted to calibrate PROBe3 or PROBe4 when the instrument is configured with two channels.

Dependencies

None

Examples

Command: `CAL:PROB1`

Query: `CAL:PROB1?`

Response: `0`

Related Commands

CALibration:PROBe[n]:RESults?

CALibration:PROBe:RESults?

TVS600A Models Only

Returns the calibration status of connected level 2 probes.

1. A value of one (1) returned by the query indicates an initialized (uncalibrated) probe.
2. A value of zero (0) indicates a calibrated probe.
3. A value of minus one (–1) indicates a probe calibration failure.
4. A value of minus two (–2) indicates a probe calibration failed because the probe was not connected to the calibration source.
5. A value of minus three (–3) indicates a calibration failed because an unsupported probe or no probe was installed.

Syntax

```
CALibration:PROBe[1]:RESults?
CALibration:PROBe2:RESults?
CALibration:PROBe3:RESults?
CALibration:PROBe4:RESults?
```

Parameters

<calibration–status>	Query response
Not applicable	<NR1> 0 1 –1 –2 –3

Reset Value

Not applicable

Errors and Events

Execution Error –241, "Hardware missing" Attempted to query PROBe3 or PROBe4 when the instrument is configured with two channels.

Dependencies

None

Examples

Command: `CAL:PROB1`

Query: `CAL:PROB1?`

Response: `0`

Related Commands CALibration:PROBe[n]

CALibration:RESults? (Query Only)

Returns the results code for the last calibration performed. The result code is the numeric identifier for the first function to fail. A value of zero is returned when there are no failures and a value of –1 indicates a calibration is in progress.

Calibration functions have numeric identifiers in the range 2000 to 2999.

Because this query is not synchronized with instrument operations, you may want to use *WAI, *OPC, or *OPC? to wait for the CAL_OPC pending flag to clear before sending the CAL:RES? query.

Syntax

```
CALibration:RESults[:CODE]?
```

Parameters

	Query response
Not applicable	<NR1> –1 Calibration in progress 0 No failures 2000 ≤ N ≤ 2999

Reset Value Not applicable

Errors and Events None

Dependencies None

Examples Query: `CAL:RES?`

Response: `0`

Related Commands CALibration
CALibration:RESults:VERBose?

CALibration:RESults:VERBose? (Query Only)

Returns an ASCII string describing the results of the last calibration performed. The returned string will include the numeric identifier for the first function that failed. When there are no failures, the query returns a zero value and a description of the last executed function. When a failure occurs, the query returns the number of the first failed function followed by detailed results information. If the query returns a value of –1, calibration is in progress.

Calibration functions have numeric identifiers in the range 2000 to 2999.

Because this query is not synchronized with instrument operations so you may want to use *WAI, *OPC, or *OPC? to wait for the CAL_OPC pending flag to clear before sending the CAL:RES? query.

Syntax

```
CALibration:RESults:VERBose?
```

Parameters

	Query response
Not applicable	<NR1>,<string> <NR1>:results code –1 (Calibration in progress) 0 (No failures) 2000 ≤ N ≤ 2999 <string>:verbose results (error specific)

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `CAL:RES:VERB?`

Response: `"2001,CAL_1 max=444 min=222 . . ."`

Related Commands

CALibration[:ALL]
CALibration:RESults?

FORMat Subsystem

This section describes the commands in the FORMat subsystem. See Figure 2–10. They are used to set the format of acquisition record data and measurement data transferred out of the waveform analyzer. The primary data transfer commands are CALCulate:DATA?, [SENse:]DATA?, and TRACe[:DATA]?.

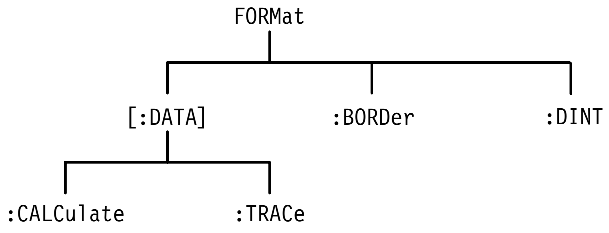


Figure 2–10: FORMat subsystem hierarchy

FORMat
FORMat?

Sets or queries the type of encoding used to transfer acquisition records acquired with the SENSe subsystem. The format applies to transfer through word serial and RS232 external interfaces. FDC transfers are always in binary format.

The number of significant digits returned by an ASCII data query is not selectable. A setting of ASCii, 0 selects a format with a varying number of significant digits depending on the origin of the data. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a definition of the data formats for <type>.

Syntax

```
FORMat[:DATA] <type>[,<length>]
FORMat[:DATA]?
```

Parameters

<type>[,<length>]	Query response
ASCii[,0] INTeger[,16]	ASC,0 INT,16

Reset Value

ASC,0

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set format to an illegal value.

Dependencies

None

Examples

Command: `FORM INT`

Query: `FORM?`

Response: `INT,16`

Related Commands

[SENSe:]DATA?
TRACe[:DATA]?
FORM:BORD

FORMat:BORDer
FORMat:BORDer?

Sets or queries the byte order used to transfer binary data through all external interfaces. The FORMat selections REAL and INTeger produce binary data. This command has no effect when FORMat is set to ASCii.

Normal is known as “Motorola order”. Swapped is known as “Intel order”.

Syntax

```
FORMat:BORDer <order>
FORMat:BORDer?
```

Parameters

<order>		Query response
NORMal	(high byte–low byte)	NORM
SWAPped	(low byte–high byte)	SWAP

Reset Value

NORM

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set byte order to an illegal value.

Dependencies

None

Examples

Command: `FORM:BORD SWAP`

Query: `FORM:BORD?`

Response: `SWAP`

Related Commands

[SENSe:]DATA?
CALCulate:DATA?
TRACe[:DATA]?
FORM
FORM:CALC
FORM:TRAC AATS
FORM:TRAC REF

FORMat:CALCulate
FORMat:CALCulate?

Sets or queries the type of data format used to transfer data produced by a CALC block. The format applies to transfer through word serial and RS232 external interfaces. The REAL type corresponds to a 32-bit Float type number in the C programming language. FDC tranfers are always in binary format.

The number of significant digits returned by an ASCII data query is not selectable. A setting of ASCii, 0 selects a format with a varying number of significant digits depending on the origin of the data. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a definition of the data formats for <type>.

Syntax

```
FORMat[:DATA]:CALCulate[n] <type>[,<length>]
FORMat[:DATA]:CALCulate[n]?
```

Parameters

<type>[,<length>]	Query response
ASCii[,0]	ASC,0
REAL,32	REAL,32

Reset Value

ASC,0

Errors and Events

Execution Error –224, “Illegal parameter value”
Attempted to set format to an illegal value.

Dependencies

None

Examples

Command: `FORM:CALC2 REAL,32`

Query: `FORM:CALC2?`

Response: `REAL,32`

Related Commands

CALCulate:DATA?
TRACe[:DATA]?
FORM:BORD

FORMat:DINTerchange
FORMat:DINTerchange?

TVS600A Models Only

Sets or queries the data interchange switch used during transfer of data through word serial and RS232 external interfaces. When ON is selected all data transfers to these interfaces are wrapped in DIF expressions. For more information DIF expressions, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

This command has no effect on FDC data transfers.

Syntax

```
FORMat:DINTerchange <boolean>
FORMat:DINTerchange?
```

Parameters

<boolean>	Query response
1 or ON 0 or OFF	1 0

Suffix Units and Multipliers

None

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command: `FORM:DINT ON`

Query: `FORM:DINT?`

Response: 1

Related Commands

[SENSe:]DATA?
CALCulate:DATA?
TRACe[:DATA]?

FORMat:TRACe:AATS
FORMat:TRACe:AATS?

Sets or queries the type of data format used to transfer the timestamp trace produced by Auto Advance acquisition. The format applies to transfer through any word serial and RS232 external interfaces. The REAL type corresponds to a Float type number in the C programming language. FDC tranfers are always in binary format.

The number of significant digits returned by an ASCII data query is not selectable. A setting of zero selects a format with a varying number of significant digits depending on the value of the data point. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a definition of the data formats for <type>.

Syntax

```
FORMat[:DATA]:TRACe:AATS <type>[,<length>]
FORMat[:DATA]:TRACe:AATS?
```

Parameters

<type>[,<length>]	Query response
ASCii[,0] REAL,32	ASC,0 REAL,32

Reset Value

ASC,0

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set format to an illegal value.

Dependencies

None

Examples

Command: `FORM:TRAC:AATS REAL,32`

Query: `FORM:TRAC:AATS?`

Response: `REAL,32`

Related Commands

TRACe[:DATA]?
FORM:BORD

FORMat:TRACe:REF
FORMat:TRACe:REF?

TVS600A Models Only

This command sets/queries the type of encoding used to transfer a Reference trace through word serial and RS232 external interfaces.

The number of significant digits returned by an ASCII data query is not selectable. A setting of ASCii, 0 selects a format with a varying number of significant digits depending on the value of the data point. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a definition of the data formats for <type>.

This command has no effect on FDC data transfers which are always REAL,32.

Syntax

```
FORMat[:DATA]:TRACe:REF <type>[,<length>]
FORMat[:DATA]:TRACe:REF?
```

Parameters

<type>,<length>	Query response
ASCii[,0] REAL,32	ASC,0 REAL,32

Suffix Units and Multipliers

None

Reset Value

ASC,0

Errors and Events

Command Error –141, “Invalid character data” or
Execution Error –224, “Illegal parameter value”
Attempted to set format to an illegal value.

Dependencies

None

Examples

Command: `FORM:TRAC:REF REAL,32`

Query: `FORM:TRAC:REF?`

Response: `REAL,32`

Related Commands

TRACe[:DATA]?
TRACe[:DATA]:AATS?

FUNCtion and DATA Subsystems

This section describes the commands in the [SENSe:]FUNCtion and [SENSe:]DATA subsystems. See Figure 2–11 and Figure 2–12. These commands control sense functions and provide a means of accessing the data produced by the sense functions.

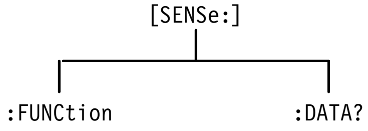


Figure 2–11: FUNCtion and DATA hierarchy

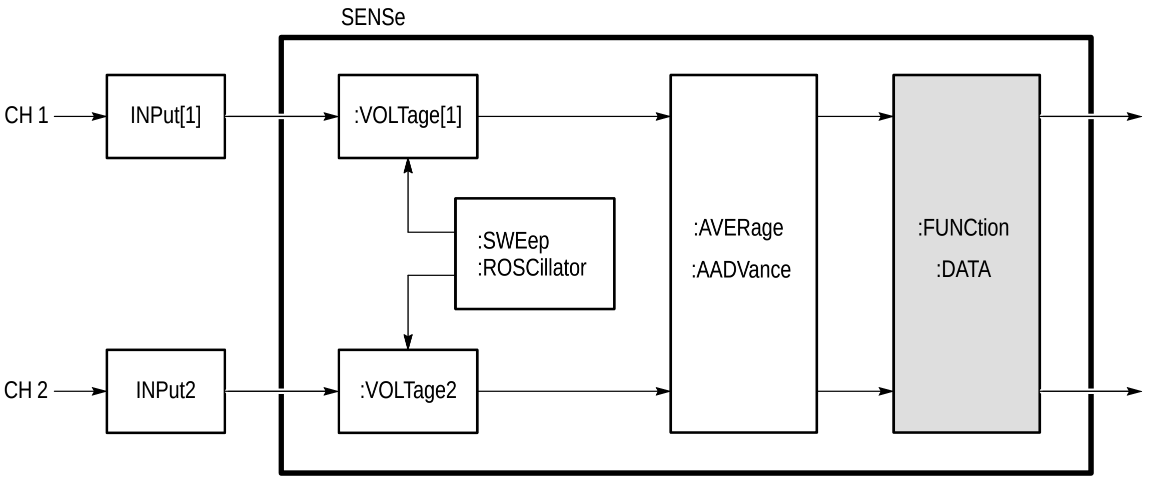


Figure 2–12: FUNCtion and DATA functional model

FUNCtion[:ON]
FUNCtion[:ON]?

Sets or queries which sense functions are enabled. XTIM:VOLT n functions acquire data from the input channel <n>. You can enable only one function at a time when FUNCtion:CONCurrent is OFF. Set FUNCtion:CONCurrent to ON if you want to enable more functions.

Channel names and function strings may be used interchangeably for parameters. This is not standard SCPI practice and does not provide horizontal compatibility.

The query FUNCtion[:ON]? always returns a comma-separated list of active sense functions. The functions are returned as a list of ASCII strings ordered by ascending channel number.

Syntax

```
[SENSe:]FUNCtion[:ON] <function>{,<function>}
[SENSe:]FUNCtion[:ON]?
```

Parameters

<function>	Query response
	<string>
“XTIMe:VOLTage[:DC] 1” or CHAN1	“XTIM:VOLT 1”
“XTIMe:VOLTage[:DC] 2” or CHAN2	“XTIM:VOLT 2”
“XTIMe:VOLTage[:DC] 3” or CHAN3	“XTIM:VOLT 3”
“XTIMe:VOLTage[:DC] 4” or CHAN4	“XTIM:VOLT 4”

Reset Value

“ ” (null string)

Errors and Events

Execution Error –221, “Settings conflict”
Attempted to enable multiple sense functions
when FUNCtion:CONCurrent is OFF.

Execution Error –224, “Illegal parameter value”
Attempted to enable an invalid sense function.

Execution Error –241, “Hardware missing”
Attempted to enable “XTIM:VOLT 3” or “XTIM:VOLT 4”
when the instrument is configured with two channels.

Dependencies

Enabling a function when FUNCtion:CONCurrent is OFF will turn off the currently enabled function.

This command determines the number of records returned by “DATA?”.

Examples

Command: `FUNC "XTIM:VOLT 1"`

Query: `FUNC?`

Response: `"XTIM:VOLT 1"`

Related Commands

FUNCtion:OFF
FUNCtion:CONCurrent
FUNCtion[:ON]:ALL
DATA?

FUNCtion[:ON]:ALL

Enables all sense functions. XTIM:VOLT n functions acquire data from the input channel <n>. You must first set FUNCtion:CONCurrent to ON before sending FUNCtion:ALL to enable all functions. To enable individual functions, use FUNCtion:[:ON].

Syntax

```
[SENSe:]FUNCtion[:ON]:ALL
```

Data Types

None

Reset Value

Not applicable

Errors and Events

Execution Error –221, “Settings conflict”
Attempted to execute this command when FUNCtion:CONCurrent is OFF.

Dependencies

This command determines the number of records returned by “DATA?”.

Examples

Command: `FUNC:ALL`

Related Commands

FUNCtion:OFF:ALL
FUNCtion:CONCurrent

FUNCtion[:ON]:COUNt? (Query Only)

Returns the number of enabled sense functions. This command is useful when FUNCtion:CONCurrent is ON because it tells you how many responses to expect when sending a FUNC[:ON]? or DATA? query.

Syntax

```
[SENSe:]FUNCtion[:ON]:COUNt?
```

Parameters

<count>	Query response
Not applicable	<NR1>

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Query: `FUNC:COUN?`

Response: 1

Related Commands

FUNCtion:OFF:COUNt?
FUNCTion:CONCurrent

FUNCtion:OFF
FUNCtion:OFF?

Sets or queries which sense functions are disabled. Only active functions provide data when you send the query DATA?. To disable all functions at once, use FUNCtion:OFF:ALL.

The query FUNCtion:OFF? always returns a comma-separated list of the disabled sense functions. The functions are returned as a list of ASCII strings ordered by ascending channel number.

Channel names and function strings may be used interchangeably for parameters. This is not standard SCPI practice and does not provide horizontal compatibility.

Syntax

```
[SENSe:]FUNCtion:OFF <function>{,<function>}
[SENSe:]FUNCtion:OFF?
```

Parameters

<function>	Query response
	<string>
“XTIMe:VOLTage[:DC] 1” or CHAN1	“XTIM:VOLT 1”
“XTIMe:VOLTage[:DC] 2” or CHAN2	“XTIM:VOLT 2”
“XTIMe:VOLTage[:DC] 3” or CHAN3	“XTIM:VOLT 3”
“XTIMe:VOLTage[:DC] 4” or CHAN4	“XTIM:VOLT 4”

Reset Value

All functions

Errors and Events

Execution Error –221, “Settings conflict”
Attempted to disable a function when FUNCtion:CONCurrent is OFF.

Execution Error –224, “Illegal parameter value”
Attempted to disable an invalid sense function.

Dependencies

This command determines the number of acquisition records returned by the query DATA?.

Examples

Command: `FUNC:OFF "XTIM:VOLT 1"`

Query: `FUNC:OFF?`

Response: `"XTIM:VOLT 1"`

Related Commands

FUNCtion[:ON]
FUNCtion:OFF:ALL
FUNCtion:CONCurrent

FUNCtion:OFF:ALL

Disables all sense functions at once without the side effects of sending the reset command *RST. Only active functions provide data when you send the query DATA?. To disable a single function, use FUNCtion:OFF.

Syntax

```
[SENSe:]FUNCtion:OFF:ALL
```

Data Types

None

Reset Value

Not applicable

Errors and Events

Execution Error –221, “Settings conflict”
Attempted to execute this command when FUNCtion:CONCurrent is OFF.

Dependencies

This command determines the number of records returned by the query DATA?.

Examples

Command: `FUNC:OFF:ALL`

Related Commands

FUNCtion[:ON]:ALL

FUNCtion:OFF:COUNt? (Query Only)

Returns the number of disabled sense functions. Use FUNCtion[:ON] to enable functions.

Syntax

```
[SENSe:]FUNCtion:OFF:COUNt?
```

Parameters

<count>	Query response
Not applicable	<NR1>

Reset Value

4

Errors and Events

None

Dependencies

None

Examples

Query: `FUNC:OFF:COUN?`

Response: 1

Related Commands

FUNCtion[:ON]:COUNt?

FUNCtion:CONCurrent
FUNCtion:CONCurrent?

Sets or queries whether more than one sense function can be enabled at a time. When you set FUNCtion:CONCurrent from OFF to ON, the state of individual functions does not change. All disabled functions remain off and if one function was enabled, it remains on. While FUNCtion:CONCurrent is ON, you can enable any and all provided functions.

When FUNCtion:CONCurrent is OFF, and you define a new function with FUNCtion[:ON], the specified function is enabled and all others are disabled.

Syntax

```
[SENSe:]FUNCtion:CONCurrent <boolean>
[SENSe:]FUNCtion:CONCurrent?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

1

Errors and Events

None

Dependencies

Setting FUNCtion:CONCurrent from ON to OFF will enable XTIM:VOLT 1 and disable all other functions.

Examples

Command: `FUNC:CONC ON`

Query: `FUNC:CONC?`

Response: 1

Related Commands

FUNCtion[:ON]
FUNCtion:OFF

FUNCtion:STATe
FUNCtion:STATe?

Sets or queries the state of a specified sense function. It operates like the FUNCtion[:ON] and FUNCtion:OFF commands which allow you to enable or disable a function.

Channel names and function strings may be used interchangeably for parameters. This is not standard SCPI practice and does not provide horizontal compatibility.

Syntax

```
[SENSe:]FUNCtion:STATe <function>, <boolean>
[SENSe:]FUNCtion:STATe? <function>
```

Parameters

<function>	Query response
“XTIMe:VOLTage[:DC] 1” or CHAN1 “XTIMe:VOLTage[:DC] 2” or CHAN2 “XTIMe:VOLTage[:DC] 3” or CHAN3 “XTIMe:VOLTage[:DC] 4” or CHAN4	Not applicable

<boolean>	Query response
<NRf>	<NR1>
1 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

Execution Error –224, “Illegal parameter value”
Attempted to to enable or disable an invalid SENSe function.

Execution Error –241, “Hardware missing”
Attempted to enable or disable “XTIM:VOLT 3” or “XTIM:VOLT 4” when the instrument is configured with two channels.

Dependencies

This command determines the number of records returned by the query DATA?.

Examples

Command: `FUNC:STAT "XTIM:VOLT 1", ON`

Query: `FUNC:STAT? "XTIM:VOLT 1"`

Response: 1

Related Commands

FUNCtion[:ON]
FUNCtion:OFF
FUNCtion:CONCurrent

DATA? (Query Only)

Returns the results for the specified function or for all sense functions that are enabled. If auto-advance acquisition is on, then DATA? returns all acquisition records through AADV:RECord:COUNt, beginning with AADV:RECord:STARt, for the specified channel. The format of the data is determined by the FORMat subsystem. The format applies to transfer through word serial and RS232 external interfaces only.

When you specify a sense function, DATA? returns only the results of the specified function. When no sense function is specified, DATA? returns the results of all sense functions that are enabled. The data is returned in ascending channel order. You can access individual auto-advance records with commands from the TRACe and CALCulate subsystems.

DATA? does not return data until pending acquisitions or calculations are complete. You can use the commands *WAI, *OPC, or *OPC? to synchronize the response in a different manner.

Channel names and function strings may be used interchangeably for parameters. This is not standard SCPI practice and does not provide horizontal compatibility.

Syntax

```
[SENSe:]DATA? [<function>]
```

Parameters

<function>	Query response
“XTIMe:VOLTage[:DC] 1” or CHAN1 “XTIMe:VOLTage[:DC] 2” or CHAN2 “XTIMe:VOLTage[:DC] 3” or CHAN3 “XTIMe:VOLTage[:DC] 4” or CHAN4	Not applicable

<data>	Query response
Not applicable	Defined by FORMat[:DATA]

Reset Value

Not applicable

Errors and Events

Execution Error –224, “Illegal parameter value”
Attempted to query data from an invalid sense function.

Execution Error –230, “Data corrupt or stale.”
Attempted to query data that is invalid, incomplete or stale.

Execution Error –241, “Hardware missing”
Attempted to query data from “XTIM:VOLT 3” or “XTIM:VOLT 4” when the instrument is configured with two channels.

Dependencies

The input channel number is used by the SENSe and TRACe subsystems For example, the command DATA? “XTIM:VOLT 1” is equivalent to TRAC[:DATA]? CHAN1 and both refer to the input channel 1.

Examples

Query: `DATA?`

Response: `<data>`

Related Commands

FUNCtion[:ON]
FUNCtion:CONCurrent
FORMat[:DATA]
DATA:PREamble?

DATA:LIST
DATA:LIST?

Refer to the TRACE:LIST command and TRACE:LIST query on page 2–224.

DATA:PREamble? (Query Only)

Returns the data preamble for the specified function or for all enabled sense functions. If auto-advance acquisition is on, then one preamble for each enabled channel is returned.

When you specify a sense function, DATA:PREamble? returns only the preamble of the specified function. When no sense function is specified, DATA:PREamble? returns the preambles of all enabled sense functions. The preambles are returned in ascending channel order.

DATA:PREamble? does not return data until pending acquisitions or calculations are complete. You can use the commands *WAI, *OPC, or *OPC? to synchronize the response in a different manner.

Channel names and function strings may be used interchangeably for parameters. This is not standard SCPI practice and does not provide horizontal compatibility.

Syntax

```
[SENSe:]DATA:PREamble? [<function>]
```

Parameters

<function>	Query response
“XTIMe:VOLTage[:DC] 1” or CHAN1 “XTIMe:VOLTage[:DC] 2” or CHAN2 “XTIMe:VOLTage[:DC] 3” or CHAN3 “XTIMe:VOLTage[:DC] 4” or CHAN4	Not applicable

<preamble>	Query response
Not applicable	DIF expression

Reset Value

Not applicable

Errors and Events

Execution Error –224, “Illegal parameter value”
Attempted to query data from an invalid sense function.

Execution Error –230, “Data corrupt or stale.”
Attempted to query data that is invalid, incomplete or stale.

Execution Error –241, “Hardware missing”
Attempted to query data from “XTIM:VOLT 3” or “XTIM:VOLT 4” when the instrument is configured with two channels.

Dependencies

The input channel number is used by the SENSe and TRACe subsystems For example, the command DATA? “XTIM:VOLT 1” is equivalent to TRAC[:DATA]? CHAN1 and both refer to the input CH 1.

Examples

Query: `DATA:PRE?`

Response:

```
DIF(VERS 1995.0 SCOP PRE)
IDEN(NAME "CHAN1" INST(NAME "TVS645A" ID "B010100"))
ENC(FORM ASC NVAL -32768 ORAN 32767 URAN -32767)
DIM=X(TYPE IMPL SCAL <nr3> OFFS <nr3> SIZE <nr1> UNIT
"S")
DIM=Y(TYPE EXPL SCAL <nr3> OFFS <nr3> SIZE <nr1> UNIT
"V")
DATA(CURV(CTYP NONE))
```

Related Commands

FUNCtion[:ON]
FUNCtion:CONCurrent
FORMat[:DATA]
DATA?

INITiate and ABORt Subsystems

This section describes the commands in the INITiate and ABORt subsystems. See Figure 2–13. These commands operate with the ARM and TRIGger subsystems to start signal acquisition.

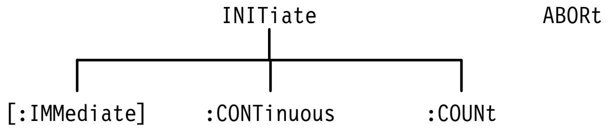


Figure 2–13: INITiate and ABORt subsystem hierarchy

INITiate

Starts the acquisition and measurement systems. INITiate causes the arm/trigger state-machine to exit the idle state and be ready for any defined ARM or TRIGger condition. This command is essential to start waveform analyzer acquisitions and measurements.

INITiate sets the ACQ_OPC pending flag. You may want to use *WAI, *OPC, or *OPC? to synchronize the command. The INIT command contains an implicit *WAI. That is, the INIT command will not begin execution until all previously received commands have been executed and all side effects have been resolved.

Following the INIT command, especially when using FDC without a separate *TRACe? query prior to reading the data, a *WAI should be sent before sending any commands which change the state of the instrument. Without the *WAI, the command may cause the acquisition task to abort before all data transfer operations are completed. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for more information.

Syntax

```
INITiate[:IMMediate]
```

Data Types

None

Reset Value

Not applicable

Errors and Events

Execution Error –213, "Init ignored"
Sent INITiate when the instrument was not in the idle state.

Dependencies

None

Examples

Command: `INIT`

Related Commands

ABORt

INITiate:CONTinuous
INITiate:CONTinuous?

Sets or queries whether the acquisition loop repeats continuously, including any defined arm or trigger conditions. Use this command only when FDC is employed to transfer data. INITiate:CONT may be difficult to interrupt once initiated.

***NOTE**. Your waveform analyzer may not respond after receiving INIT:CONT. To end acquisition and return control, send the VXIbus word serial command <Abort Normal Operation> then INIT:CONT OFF.*

When INIT:CONTinuous is set to ON and you send a self-synchronizing command, such as a TRACe? query, the parser does not execute subsequent commands until the current command (INIT:CONT) completes and an OPC occurs. OPC cannot occur until INIT:CONT terminates. When a defined trigger event or arm event does not occur, INIT:CONT does not terminate. Unless you have sent a data query or other self-synchronizing command, you can recover control by sending the *RST command.

When you send INIT:CONT, acquisition continues until the instrument receives an *RST, or VXI word serial <Abort Normal Operation> or <End Normal Operation> command and INIT:CONT OFF. The command *RST is effective only when you have not sent a self-synchronizing command, such as TRACe?. You can use the command ABORt to stop the current acquisition and automatically restart continuous acquisition.

The setting for INIT:COUNt is ignored when INIT:CONTinuous is set to ON.

Syntax

```
INITiate:CONTinuous <boolean>
INITiate:CONTinuous?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

Setting INIT:CONTinuous to ON performs an implicit INITiate[:IMMediate] and, if the Fast Data Channel is active, sends a continuous stream of measurement results.

Examples

Command: `INIT:CONT ON`

Query: `INIT:CONT?`

Response: 1

Related Commands

INITiate
ABORt

INITiate:COUNt
INITiate:COUNt?

Sets or queries the number of times to repeat the arm/trigger acquisition loop. This command is most useful when the FDC is employed to transfer data.

For acquisition looping using FDC, you might set COUNt to a value of 10. The instrument then cycles through ten arm/trigger loops, acquiring signals from all active channels during each pass through the loop. The specified waveform records and measurements are transferred at the end of each cycle. Upon completion of the tenth loop the arm/trigger state machine returns to the idle state. Without FDC, only the waveform record acquired on the last cycle and measurements on it are available.

If you have defined a Fast Data Channel, setting COUNt to a value other than one will cause the waveform analyzer to send multiple measurement results, one for each acquisition.

Syntax

```
INITiate:COUNt <count>
INITiate:COUNt?
```

Parameters

<count>	Query response
<NRf>	<NR1>
1 ≤ N ≤ 10000	
MINimum	1
MAXimum	10000

Reset Value

1

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set count to an illegal value.

Dependencies

When INIT:CONTinuous is set to ON, INITiate:COUNt is ignored.

Examples

Command: `INIT:COUN 10`

Query: `INIT:COUN?`

Response: `10`

Related Commands

INITiate
INITiate:CONTinuous
ABORt

ABORt

Stops all acquisition and measurements and returns the arm/trigger subsystem to the idle state. Any remaining data is corrupt. When INITiate:CONTinuous is set to ON, the current acquisition is terminated and acquisition is restarted.

Syntax

```
ABORt
```

Data Types

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

Execution Error –230, "Data corrupt or stale."
Sent the command DATA? after sending ABORt.

Examples

Command: `ABOR`

Related Commands

INITiate

INPut Subsystem

This section describes each command and query in the INPut subsystem. The INPut subsystem controls the parameters shown in Figure 2–14. Figure 2–15 shows the part of the waveform analyzer controlled by the INPut commands. The input channel number, defined at the probe connector, is shared as the parameter <n> for the INPut<n> and VOLTage<n> commands. Only the four-channel waveform analyzers will accept INPut3 and INPut4 commands.

This section describes each command and query in the INPut subsystem. The INPut subsystem controls the parameters shown in Figure 2–14. Figure 2–15 shows the part of the waveform analyzer controlled by the INPut commands. The input channel number, defined at the probe connector, is shared as the parameter <n> for the INPut<n> and VOLTage<n> commands. Only the four-channel waveform analyzers will accept INPut3 and INPut4 commands.

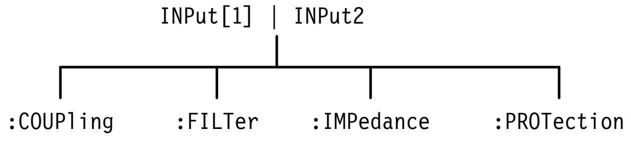


Figure 2–14: INPut subsystem hierarchy

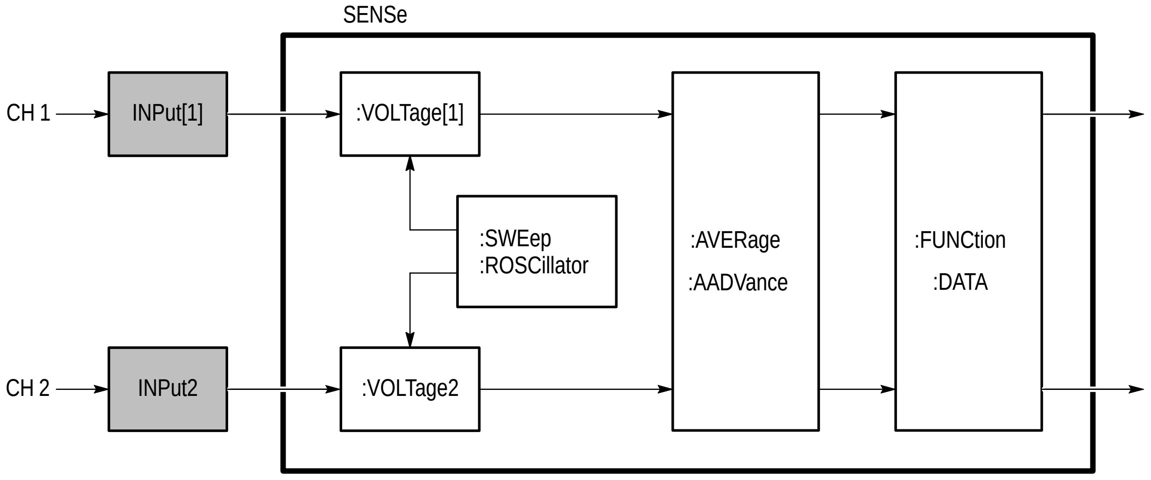


Figure 2–15: INPut subsystem functional model

INPut:COUPling
INPut:COUPling?

Sets or queries the type of signal coupling for the specified input channel. When AC coupled and INPut:IMPedance is set to 1 MΩ, frequencies below 10 Hz are attenuated. When AC coupled and INPut:IMPedance is set to 50 Ω, frequencies below 200 kHz are attenuated. Setting COUPling to GROund connects the input of the amplifier to ground and presents a high impedance to the incoming signal.

Syntax

```
INPut<n>:COUPling <coupling>
INPut<n>:COUPling?
```

Parameters

<n> (Channel Number)[1]	Query response
1 2 3 4	Not applicable

[1] **The input channel number <n> is used by the commands INPut<n> and VOLTage<n>. If you omit <n>, the default is channel 1.**

<coupling>	Query Response
AC DC GROund	AC DC GRO

Reset Value

DC

Errors and Events

Execution Error –141, “Invalid character data, CALC1:WMC FTM”
Attempted to set coupling to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to program INPut3 or INPut4 on an
instrument configured with two channels.

Dependencies

None

Examples

Command: `INP1:COUP AC`

Query: `INP1:COUP?`

Response: `AC`

Related Commands

None

INPut:FILTer
INPut:FILTer?

Sets or queries whether the low-pass analog noise filter is on or off for the specified input channel. INPut:FILTer must be off to use the full bandwidth of the waveform analyzer. To set the limit frequency of the low pass filter, use the command INPut:FILTer:FREQuency.

Syntax

```
INPut<n>:FILTer[:LPASs][:STATe] <boolean>
INPut<n>:FILTer[:LPASs][:STATe]?
```

Parameters

<n> (Channel Number)[1]	Query response
1 2 3 4	Not applicable

[1] The input channel number <n> is used by the commands INPut<n> and VOLTage<n>. If you omit <n>, the default is channel 1.

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NF1> 1 0

Reset Value

0 (OFF)

Errors and Events

Execution Error –241, “Hardware missing”
Attempted to program INPut3 or INPut4 on an instrument configured with two channels.

Dependencies

None

Examples

Command: `INP1:FILT ON`

Query: `INP1:FILT?`

Response: 1

Related Commands INPut:FILTer:FREQuency

INPut:FILTer:FREQuency
INPut:FILTer:FREQuency?

Sets or queries the frequency limit of the low-pass analog noise filter for the specified input channel. You must enable the low pass filter with the command INPut:FILTer before the filter is effective. The filter selections are 20 MHz and 250 MHz. To use the full bandwidth of the waveform analyzer set INPut:FILTer to OFF.

Syntax

```
INPut<n>:FILTer[:LPASs]:FREQuency <frequency>
INPut<n>:FILTer[:LPASs]:FREQuency?
```

Parameters

<n> (Channel Number)[1]	Query response
1 2 3 4	Not applicable

[1] The input channel number <n> is used by the commands INPut<n> and VOLTage<n>. If you omit <n>, the default is channel 1.

<frequency> (Limit of Low Pass Filter) [1]	Query response
<NRf>	<NF3>
20E6	20.0E+6
250E6[2]	250.0E+6
MINimum	20.0E+6
MAXimum	250.0E+6

[1] The default multiplier for the <frequency> parameter is HZ. You can also use KHZ or MHZ.

[2] When using the 250 MHz filter on the TVS621A or TVS641A input channels, the effective system bandwidth is typically 180 MHz.

Reset Value 250.0E+6

Errors and Events Execution Error –222, “Data out of range”
Attempted to set low pass frequency to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to program INPut3 or INPut4 on an instrument configured with two channels.

Dependencies None

Examples

Command: `INP1:FILT:FREQ 20E6`

Query: `INP1:FILT:FREQ?`

Response: `20.0E+6`

Related Commands INPut:FILTer

INPut:IMPedance
INPut:IMPedance?

Sets or queries the input impedance of the specified input channel. The impedance selections are 50 Ω and 1 MΩ. When input protection is enabled, an overload of the 50 Ω input automatically sets impedance to 1 MΩ.

Syntax

```
INPut<n>:IMPedance <impedance>
INPut<n>:IMPedance?
```

Parameters

<n> (Channel Number)[1]	Query response
1 2 3 4	Not applicable

[1] **The input channel number <n> is used by the commands INPut<n> and VOLTage<n>. If you omit <n>, the default is channel 1.**

<impedance> (Channel impedance)[1]	Query response
<NRf>	<NF3>
50	50.0E+0
1E6	1.0E+6
MINimum	50.0E+0
MAXimum	1.0E+6

[1] **The default multiplier for the <impedance> parameter is OHM. You can also use KOHM or MOHM.**

Reset Value

1.0E+6

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set impedance to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to program INPut3 or INPut4 on an instrument configured with two channels.

Dependencies

None

Examples

Command: `INP1:IMP 1E6`

Response: `INP1:IMP?`

Query: `1.0E+6`

Related Commands

INPut:PROTection:STATe

INPut:PROBe:ATTenuation?

Queries the attenuation ratio of a connected probe.

A value of 1 indicates that a 1X probe is attached or that no probe is attached.

Syntax

```
INPut[1]:PROBe:ATTenuation?
INPut2:PROBe:ATTenuation?
INPut3:PROBe:ATTenuation?
INPut4:PROBe:ATTenuation?
```

Parameters

<attenuation factor>	Query response
Not applicable	<NR1>

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

Execution Error –241, “Hardware missing”
Attempted to query INPut3 or INPut4 when the instrument is configured with two channels.

Dependencies None

Examples Query: `INP1:PROB:ATT?`

Response: `10`

Related Commands INPut:PROBe:IDENt?

INPut:PROBe:IDENtification?

TVS600A Models Only

Returns the identification data of a connected probe.

Identification data for level 2 probes will consist of probe type.

A null string will be returned for level 1 probes or when no probe is connected.

Syntax

```
INPut[1]:PROBe:IDENtification?
INPut2:PROBe:IDENtification?
INPut3:PROBe:IDENtification?
INPut4:PROBe:IDENtification?
```

Parameters

<string>	Query response
Not applicable	<string>

Suffix Units and Multipliers None

Reset Value Not applicable

Errors and Events Execution Error –241, “Hardware missing”
Attempted to query INPut3 or INPut4 when the
instrument is configured with two channels.

If an error occurs while reading the probe, then the query will return the string “ERROR”.

Dependencies None

Examples Query: `INP1:PROB:IDEN?`

Response: `"P6245"`

Related Commands

INPut:PROBe:ATTn?
INPut:PROBe:OFFset?

INPut:PROBe:OFFSet?

TVS600A Models Only

Queries the offset scale of a connected probe.

A value of 0 indicates that a 1X probe is attached or that no probe at all is attached.

Syntax

```
INPut[1]:PROBe:OFFSet?
INPut2:PROBe:OFFSet?
INPut3:PROBe:OFFSet?
INPut4:PROBe:OFFSet?
```

Parameters

<offset scale>	Query response
Not applicable	<NR3>

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

Execution Error –241, “Hardware missing”
Attempted to query INPut3 or INPut4 when the
instrument is configured with two channels.

Dependencies

None

Examples

Query: `INP1:PROB:OFFS?`

Response: `10.0`

Related Commands

INPut:PROBe:IDENt?
INPut:PROBe:ATTenuation?

INPut:PROTection:STATe
INPut:PROTection:STATe?

Sets or queries the state of the input protection circuitry for *all* input channels. When protection is enabled (ON), an input overload will automatically set the input channel impedance to 1 MΩ in order to protect the input.

Because the waveform analyzer stores the INPut:PROTection:STATe in nonvolatile RAM, this setting is recalled at power-up. However, it is not saved with a stored settings, and it is not changed by the reset command *RST. The factory default is to enable input protection.

CAUTION. *Use caution when disabling input protection. Damage may occur to the instrument input channels if the overload condition persists or is extreme.*

Syntax

```
INPut:PROTection:STATe <boolean>
INPut:PROTection:STATe?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NF1> 1 0

Reset Value

Not applicable

Errors and Events

Execution Error –203, “Command protected”
Attempted to change the protection without first removing the restriction with the SYSTem:PROTect command.

Device Specific Error –310, “System error”
An input overload generates a device-specific error whether or not input protection is enabled.

Dependencies

Although other INPut commands separately control each channel, input protection for all channels is enabled or disabled at the same time. Disabling the protection for one channel automatically disables the protection for all channels.

Examples

Command:	`INP:PROT:STAT ON`
Query:	`INP:PROT:STAT?`
Response:	1

Related Commands

INPut:IMPedance
SYStem:PROTect

MEMory Subsystem

This section describes the commands in the MEMory subsystem. See Figure 2–16. These commands store and retrieve instrument settings. The waveform analyzer can store the current instrument state in any of ten nonvolatile memory locations or in a file on your controller.

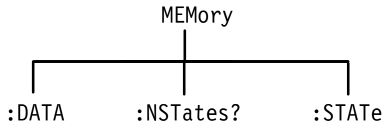


Figure 2–16: MEMory subsystem hierarchy

MEMory:DATA
MEMory:DATA?

Sets or queries the instrument settings (or state) for the ten on-board nonvolatile memory locations. With the query form, you can save the current settings (SAV0) or one or more of the on board settings (SAV1–10) to a file on your controller. Later, you can load these settings into the on board storage locations to run a suite of measurements based on the different instrument configurations. Use the command *RCL to use an on board instrument setting.

You cannot load a settings file into SAV0 because it is the current or active instrument settings. You can use the command SYSTem:SET to modify the current settings with a settings file.

Settings data is transferred in binary format, designated <arbitrary_block_data>, that is unique to the waveform analyzer.

Syntax

```
MEMory:DATA <setting>, <data>
MEMory:DATA? <setting>
```

Parameters

<source>	Query response
SAV0 SAV1 SAV2 . . SAV10	Not applicable

<data>	Query response
<arbitrary_block_data>	<arbitrary_block_data>

Reset Value

Not applicable

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set or query an invalid setting number.

Execution Error –233, “Invalid version”
Attempted to load a block of binary settings that have a different version number than the instrument.

Dependencies

None

Examples

Command:	`MEM:DATA SAV1,<arb_block_data>`
Query:	`MEM:DATA? SAV1`
Response:	`<arb_block_data>`

Related Commands

MEMory:DATA
MEMory:NSTates?
MEMory:STATe:CATalog?
MEMory:STATe:DEFine?
SYSTem:SET
*RCL
*SAV

MEMory:NSTates? (Query Only)

Returns the number of instrument settings (states) that can be stored in the waveform analyzer. Settings locations can be used or filled with the command *SAV and MEMory:DATA. Once stored, settings can be recalled with the command *RCL.

Syntax

```
MEMory:NSTates?
```

Parameters

<states>	Query response
Not applicable	<NR1> 10

Reset Value

10

Errors and Events

None

Dependencies

None

Examples

Query: `MEM:NST?`

Response: `10`

Related Commands

MEMory:DATA?
MEMory:STATe:CATalog?
MEMory:STATe:DEFine?
SYSTem:SET
*RCL
*SAV

MEMory:STATe:CATalog? (Query Only)

Returns the list of predefined names for the on-board stored settings. The list is composed of names as quoted strings separated by commas. Settings locations can be used or filled with the commands *SAV and MEMory:DATA. Once stored, settings can be recalled with the command *RCL.

Syntax

```
MEMory:STATe:CATalog?
```

Parameters

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `MEM:STAT:CAT?`

Response: `"SAV1,SAV2,SAV3,. . . ,SAV10"`

Related Commands

MEMory:DATA
MEMory:NSTates?
MEMory:STATe:CATalog?
SYSTem:SET
*RCL
*SAV

MEMory:STATe:DEFine? (Query Only)

Returns the register number of a specified instrument settings location in the waveform analyzer. Settings locations can be used or filled with the commands *SAV and MEMory:DATA. Once stored, settings can be recalled with the command *RCL.

Syntax

```
MEMory:STATe:DEFine? <setting>
```

Parameters

<setting>	Query response
SAV0	<NR1>
SAV1	1
SAV2	2
.	3
.	.
.	.
SAV10	10

Reset Value

Not applicable

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to query an invalid setting.

Dependencies

None

Examples

Query: `MEM:STAT:DEF? SAV3`

Response: `3`

Related Commands

MEMory:DATA?
MEMory:NSTates?
MEMory:STATe:CATalog?
SYSTem:SET
*RCL
*SAV

OUTPut Subsystem

This section describes the commands in the OUTPut subsystem. See Figure 2–17. These commands route signals to the VXIbus trigger lines and enable the probe compensation signals and reference signals.

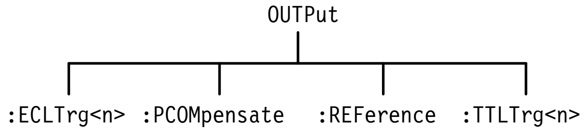


Figure 2–17: OUTPut subsystem hierarchy

OUTPut:ECLTrg<n>
OUTPut:ECLTrg<n>?

Sets or queries whether the instrument should drive (source) the specified VXIbus ECL trigger line when a trigger event occurs. You can choose inverted output polarity on the ECL trigger lines.

The waveform analyzer lets you to define the source to drive each VXIbus ECL trigger line. You can source and sense an ECL trigger line.

Syntax

```
OUTPut:ECLTrg<n>[:STATe] <boolean>
OUTPut:ECLTrg<n>[:STATe]?
```

Parameters

<n> (number of VXIbus trigger signal)	Query response
<NR1> 0 or 1	<NR1>

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command: `OUTP:ECLT0 ON`

Query: `OUTP:ECLT0?`

Response: 1

Related Commands

OUTPUt:ECLTrg<n>:SOURce
OUTPUt:ECLTrg<n>:POLarity

OUTPut:ECLTrg<n>:POLarity
OUTPut:ECLTrg<n>:POLarity?

Sets or queries the drive polarity for the specified VXIbus ECL trigger line. Normally, the ECL trigger lines are asserted in the logical high state. With INVerted polarity the trigger lines are asserted in the logical low state.

You can enable each ECL trigger line separately with the command OUTPut:ECLTrg<n>. Use the command OUTPut:ECLTrg<n>:SOURce to control the source driver for the ECL trigger lines.

Syntax

```
OUTPut:ECLTrg<n>:POLarity<polarity>
OUTPut:ECLTrg<n>:POLarity?
```

Parameters

<n> (Line number)	Query response
0 1	Not applicable

<polarity>	Query response
NORMal (active high) INVerted (active low)	NORM INV

Reset Value

NORM

Errors and Events

Execution Error –141, “Invalid character data”.
Attempted to set polarity to an illegal value.

Dependencies

Examples

Command: `OUTP:ECLT0:POL INV`

Query: `OUTP:ECLT0:POL?`

Response: `INV`

Related Commands

OUTPut:ECLTrg<n>
OUTPut:ECLTrg<n>:SOURce

OUTPut:ECLTrg<n>:SOURce
OUTPut:ECLTrg<n>:SOURce?

Sets or queries the waveform analyzer source used to drive each VXIbus ECL trigger line.

- ARM — a valid ARM event occurs which enables the TRIG:A circuit
- ATR — a valid trigger event from the TRIGger A subsystem
- BTR — a valid trigger event from the TRIGger B subsystem
- OPC — the signal indicating the active command is complete. It is derived from the Operation Complete bit in the Standard Event Status Register.
- CALC — a true evaluation of the TRG function in a CALC expression has occurred

The waveform analyzer can simultaneously source and sense an ECL trigger line.

Syntax

```
OUTPut:ECLTrg<n>:SOURce <source>
OUTPut:ECLTrg<n>:SOURce?
```

Parameters

<n> (Line number)	Query response
0 1	Not applicable

<source>	Query response
ARMed	ARM
ATRigger	ATR
BTRigger	BTR
OPC	OPC
CALC	CALC

Reset Value

ATR (ECLTrg1)
BTR (ECLTrg0)

Errors and Events

Execution Error –141, “Invalid character data”.
Attempted to set source to an illegal value.

Dependencies

None

Examples

Command: `OUTP:ECLT0:SOUR BTR`

Query: `OUTP:ECLT0:SOUR?`

Response: `BTR`

Related Commands

OUTPut:ECLTrg<n>
OUTPut:ECLTrg<n>:POLarity

OUTPut:PCOMpensate OUTPut:PCOMpensate?

Sets or queries whether the probe compensation signal is output on the front-panel connector PROBE COMPENSATION. Use the command OUTPut:PCOMpensate:FUNCtion to select the type of compensation signal. You can select either CLOCk at 1 kHz and 500 mV peak-to-peak amplitude or VOLT with a 500 mV DC level.

When compensating probes, use the BNC-to-probe tip adapter recommended by the probe manufacturer and connect the probe ground lead to signal ground.

Syntax

```
OUTPut:PCOMpensate[:STATe] <boolean>
OUTPut:PCOMpensate[:STATe]?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command: `OUTP:PCOM ON`

Query: `OUTP:PCOM?`

Response: 1

Related Commands

OUTPut:PCOMpensate:FUNC

OUTPut:PCOMpensate:FUNCtion
OUTPut:PCOMpensate:FUNCtion?

Sets or queries the source for the probe compensation signal which is output on the front-panel connector PROBE COMPENSATION. Use the command OUTPut:PCOMpensate to enable the signal. You can select either CLOCk at 1 kHz and 500 mV peak-to-peak amplitude or VOLT with a 500 mV DC level.

When compensating probes, use the BNC-to-probe tip adapter recommended by the probe manufacturer and connect the probe ground lead to signal ground.

Syntax

```
OUTPut:PCOMpensate:FUNCtion <function>
OUTPut:PCOMpensate:FUNCtion?
```

Parameters

<function>	Query response
CLOCk	CLOC
VOLTage	VOLT

Reset Value

CLOC

Errors and Events

Execution Error –141, “Invalid character data”.
Attempted to set function to an illegal value.

Dependencies

None

Examples

Command: `OUTP:PCOM:FUNC VOLT`

Query: `OUTP:PCOM:FUNC?`

Response: `VOLT`

Related Commands

OUTPut:PCOMpensate

OUTPut:REFerence
OUTPut:REFerence?

Sets or queries whether a reference signal is output on the front-panel connector REFERENCE OUTPUT. The reference signal is intended for use during instrument verification. You can direct either of two signals to the REFERENCE OUTPUT: the internal 10 MHz reference clock with a 1 V peak-to-peak amplitude or the 8.0 V DC level. Use OUTPut:REFerence:FUNCtion to select which signal appears on the output. The reference signal is turned off when you start an acquisition or reset the instrument.

Syntax

```
OUTPut:REFerence[:STATe] <boolean>
OUTPut:REFerence[:STATe]?
```

Parameters

<boolean>	Query response
<NRf>	<NR1>
1 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command: `OUTP:REF ON`

Query: `OUTP:REF?`

Response: `1`

Related Commands

OUTPut:REFerence:FUNCtion

OUTPut:REFerence:FUNCtion
OUTPut:REFerence:FUNCtion?

Sets or queries which reference signal is output on the connector REFERENCE OUTPUT. The reference signal is intended for use during instrument verification. You can direct either of two signals to the REFERENCE OUTPUT: the internal 10 MHz reference clock with a 1 V peak-to-peak amplitude or the 8.0 V DC level. Use OUTPut:REFerence to enable the signal output. The reference signal output is disabled when you start acquisition or reset the instrument.

Syntax

```
OUTPut:REFerence:FUNCtion <function>
OUTPut:REFerence:FUNCtion?
```

Parameters

<function>	Query response
CLOCk	CLOC
VOLTage	VOLT

Reset Value

CLOC

Errors and Events

Execution Error –141, “Invalid character data”.
Attempted to set the calibration reference function to an illegal value.

Dependencies

None

Examples

Command: `OUTP:REF:FUNC VOLT`

Query: `OUTP:REF:FUNC?`

Response: `VOLT`

Related Commands

OUTPut:REFerence

OUTPut:TTLTrg<n>
OUTPut:TTLTrg<n>?

Sets or queries whether the instrument should drive (source) the specified VXIbus TTL trigger lines when a trigger event occurs. You can enable each TTL trigger line separately. Use the command OUTPut:TTLTrg<n>:SOURce to pick the source for each TTL trigger line. Use the command OUTPut:TTLTrg<n>:POLarity to set the polarity of the output.

Syntax

```
OUTPut:TTLTrg<n>[:STATe] <boolean>
OUTPut:TTLTrg<n>[:STATe]?
```

Parameters

<n> (number of TTLTrg signal)	Query response
<NR1> 0,1, 2, 3, 4, 5, 6, or 7	<NR1>

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command: `OUTP:TTLT0 ON`

Query: `OUTP:TTLT0?`

Response: 1

Related Commands

OUTPut:TTLTrg<n>:POLarity
OUTPut:TTLTrg<n>:SOURce?

OUTPut:TTLTrg<n>:POLarity
OUTPut:TTLTrg<n>:POLarity?

Sets or queries the drive polarity for each VXI TTLTrg trigger line. The TTL trigger lines are normally active low. With INVerted polarity (active high), a controller can detect when all cards in a mainframe have asserted a trigger line.

Use the command OUTPut:TTLTrg<n>:SOURce to control the source driver for the TTL trigger lines. Use the command OUTPut:TTLTrg<n> to enable each TTLTrg driver.

Syntax

```
OUTPut:TTLTrg<n>:POLarity<polarity>
OUTPut:TTLTrg<n>:POLarity?
```

Parameters

<n> (number of TTLTrg signal)	Query response
<NR1> 0,1, 2, 3, 4, 5, 6, or 7	<NR1>

<polarity>	Query response
NORMal (active low) INVerted (active high)	NORM INV

Reset Value

NORM

Errors and Events

Execution Error –141, “Invalid character data”.
Attempted to set polarity to an illegal value.

Dependencies

None

Examples

Command: `OUTP:TTLT0:POL INV`

Query: `OUTP:TTLT0:POL?`

Response: `INV`

Related Commands

OUTPut:TTLTrg
OUTPut:TTLTrg<n>:SOURce?

OUTPut:TTLTrg<n>:SOURce
OUTPut:TTLTrg<n>:SOURce?

Sets or queries the waveform analyzer source for each VXI TTL trigger line. The waveform analyzer drives the VXIbus trigger lines only when they are enabled and a valid trigger event occurs. The available waveform analyzer sources are as follows:

- ARM — a valid ARM event occurs which enables the TRIG:A circuit
- ATR — a valid trigger event from the TRIGger A subsystem
- BTR — a valid trigger event from the TRIGger B subsystem
- OPC — the signal indicating the active command is complete. It is derived from the Operation Complete bit in the Standard Event Status Register.
- CALC — a true evaluation of the TRG function in a CALC expression has occurred

Use the command OUTPut:TTLTrg<n> to enable each VXI TTLTrg line.

The waveform analyzer can simultaneously source and sense TTLTrg lines.

Syntax

```
OUTPut:TTLTrg<n>:SOURce <source>
OUTPut:TTLTrg<n>:SOURce?
```

Parameters

<n> (number of TTLTrg signal)	Query response
<NR1> 0,1, 2, 3, 4, 5, 6, or 7	<NR1>

<source>	Query response
ARMed ATRigger BTRigger OPC CALC	ARM ATR BTR OP CALC

Reset Value

ARM (TTLTrg0)	ARM (TTLTrg4)
ATR (TTLTrg1)	ATR (TTLTrg5)
BTR (TTLTrg2)	BTR (TTLTrg6)
OPC (TTLTrg3)	OPC (TTLTrg7)

Errors and Events

Execution Error –141, “Invalid character data”.
Attempted to set source to an illegal value.

Dependencies

None

Examples

Query: `OUTP:TTLT0:SOUR ARM`

Query: `OUTP:TTLT0:SOUR?`

Response: `ARM`

Related Commands

OUTPut:TTLTrg
OUTPut:TTLTrg<n>:POLarity

ROSCillator Subsystem

This section describes the commands in the SENSe:ROSCillator subsystem. See Figure 2–18 and Figure 2–19. These commands control the source of the reference oscillator (clock) for the :SWEep subsystem.

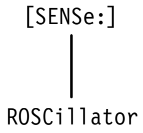


Figure 2–18: ROSCillator subsystem

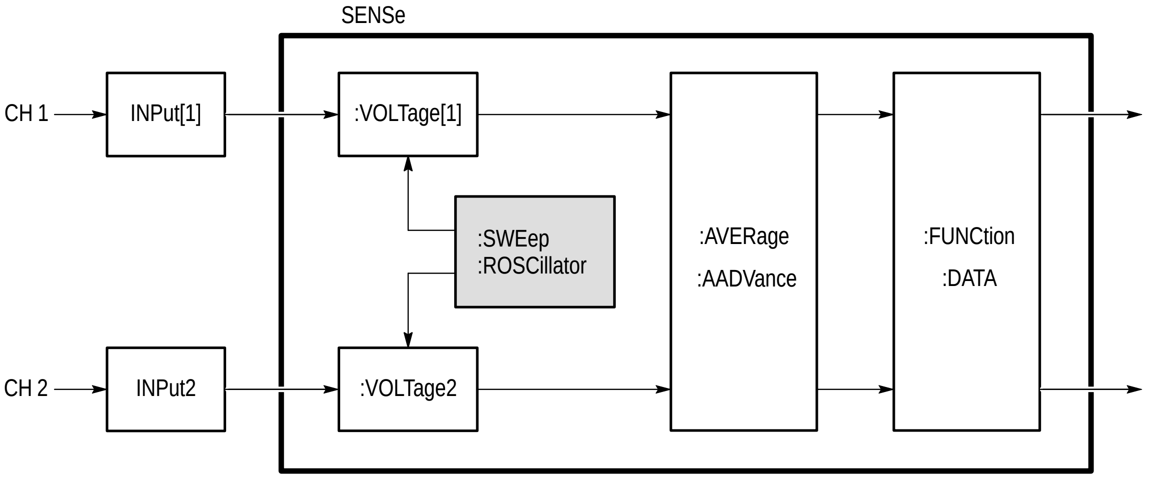


Figure 2–19: ROSCillator subsystem functional model

ROSCillator:SOURce
ROSCillator:SOURce?

Sets or queries the source of the 10 MHz clock reference for the acquisition system. The choices are CLK10 which specifies the VXIbus 10 MHz reference, and INTernal which specifies the internal waveform analyzer oscillator.

Syntax

```
[SENSe:]ROSCillator:SOURce <source>
[SENSe:]ROSCillator:SOURce?
```

Parameters

<source>	Query response
CLK10	CLK10
INTernal	INT

Reset Value

INT

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set source to an illegal value.

Dependencies

None

Examples

Command: `ROSC:SOUR CLK10`

Query: `ROSC:SOUR?`

Response: `CLK10`

Related Commands

SWEep:TINTerval

STATus Subsystem

This section describes the commands in the STATus subsystem. See Figure 2–20. These commands, along with several IEEE 488.2 Common Commands, control the status and event reporting system. The STATus subsystem provides a way to determine the state of the waveform analyzer and control what events can interrupt the system controller. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

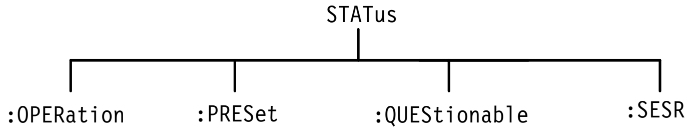


Figure 2–20: STATus subsystem hierarchy

STATus:OPERation? (Query Only)

Returns the contents of the Operation Status Register as a decimal number. The Operation Status Register, described in Table 2–2, identifies normal events, such as acquisition, that are in progress. Use the Operation Status Enable Register to determine which Operation Status events can set the Operation bit (bit 7) in the Status Byte Register.

Table 2–2: The Operation Status Register

Bit	Decimal Value	Function
0	1	**Calibrating** shows that a calibration routine is in progress.
1–3		Not used.
4	16	**Measuring/Acquiring** shows that measurement or acquisition is in progress.
5	32	**Waiting for Trigger** shows that the acquisition system is armed and waiting for a trigger event.
6	64	**Waiting for Arm** shows that the acquisition system has been initialized with INIT and is waiting to be armed.
7		Not used.
8	256	**Testing** shows that a self test routine is in progress.
9	512	**CH 1 Probe** shows that a probe is attached to the Channel 1 input.
10	1024	**CH 2 Probe** shows that a probe is attached to the Channel 2 input.
11	2048	**CH 3 Probe** shows that a probe is attached to the Channel 3 input.
12	4096	**CH 4 Probe** shows that a probe is attached to the Channel 4 input.
13–15		Not used.

Syntax

```
STATus:OPERation[:EVENt]?
```

Parameters

<event>	Query Response
Not applicable	<NR1>

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `STAT:OPER?`

Response: `512`
This response means that the only registered Operation is a probe connected to input Channel 1.

Related Commands

STATus:OPERation:CONDition?
STATus:OPERation:ENABle
STATus:OPERation:PTRansition
STATus:OPERation:NTRansition
STATus:OPERation:QENable:NTRansition
STATus:OPERation:QENable:PTRansition

STATus:OPERation:CONDition? (Query Only)

Returns the contents of the Operation Status Condition Register (OSCR). The OSCR bits correspond to the Operation Status Register bits. The query response from the Operation Status Condition Register gives the current state of the Operation event lines prior to the Operation Transition Registers. The query returns the current setting as a decimal number whose bits correspond to the Operation Status Register bits. Refer to Table 2–2 on page 2–152 for a definition of the events associated with the Operation Status Register bits. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Note that reading this register does not clear it and unused bits always return zero.

Syntax

```
STATus:OPERation:CONDition?
```

Parameters

<condition>	Query Response
Not applicable	<NR1>

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `STAT:OPER:COND?`

Response: 1

Related Commands

STATus:OPERation?
STATus:OPERation:ENABle

STATus:OPERation:ENABle
STATus:OPERation:ENABle?

Sets or queries the contents of the Operation Status Enable Register (OSER). The OSER allows you to individually disable any of the Operation events from setting bit 7 in the Status Byte Register. The query returns the current setting as a decimal number whose bits correspond to the Operation Status Register bits. Refer to Table 2–2 on page 2–152 for a definition of the events associated with the Operation Status Register bits. Setting unused enable bits does not generate an error, they are ignored. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to zero (0) which disables all Operation events.

Syntax

```
STATus:OPERation:ENABle <mask>
STATus:OPERation:ENABle?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the enable register to an illegal value.

Dependencies

None

Examples

Command: `STAT:OPER:ENAB #H0010`
This command, (decimal 16), enables only the Acquiring/Measuring event.

Query: `STAT:OPER:ENAB?`

Response: `16`

Related Commands

STATus:OPERation?
STATus:OPERation:CONDition?
STATus:PRESet

STATus:OPERation:NTRansition
STATus:OPERation:NTRansition?

Sets or queries the contents of the Operation Negative Transition Register (ONTR). When you set an event bit true (1) in the ONTR, the ONTR reports an event in the Operation Status Register when the Operation Status Condition Register event line changes from true to false (1 to 0). The query returns the current setting as a decimal number whose bits correspond to the Operation Status Register bits. Refer to Table 2–2 on page 2–152 for a definition of the events associated with the Operation Status Register bits. The command STATus:OPERation:PTRansition provides similar control for positive event transitions. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to zero (0) which disables reporting negative event transitions. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:OPERation:NTRansition <mask>
STATus:OPERation:NTRansition?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the transition register to an illegal value.

Dependencies

None

Examples

Command: `STAT:OPER:NTR #H0010`
This command, (decimal 16), sets the Operation Negative Transition Register to set the Acquiring/Measuring event bit when the current measurement or acquisition completes.

Query: `STAT:OPER:NTR?`

Response: `16`

Related Commands

STATus:OPERation?
STATus:OPERation:CONDition?
STATus:OPERation:PTRansition
STATus:PRESet

STATus:OPERation:PTRansition
STATus:OPERation:PTRansition?

Sets or queries the contents of the Operation Positive Transition Register (OPTR). When you set an event bit true (1) in the OPTR, the OPTR reports an event in the Operation Status Register when the Operation Status Condition Register event line changes from false to true (0 to 1). The query returns the current setting as a decimal number whose bits correspond to the Operation Status Register bits. Refer to Table 2–2 on page 2–152 for a definition of the events associated with the Operation Status Register bits. The command STATus:OPERation:NTRansition provides similar control for negative event transitions. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to one (1) which enables reporting positive event transitions. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:OPERation:PTRansition <mask>
STATus:OPERation:PTRansition?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the transition register to an illegal value.

Dependencies

None

Examples

Command: `STAT:OPER:PTR #H0010`
With this parameter value, (decimal 16), the Operation Positive Transition Register sets *only* the Acquiring/Measuring event bit

when measurement or acquisition starts. All other positive Operation events are disabled.

Query: `STAT:OPER:PTR?`

Response: `16`

Related Commands

STATus:OPERation?
STATus:OPERation:CONDition?
STATus:OPERation:NTRansition
STATus:PRESet

STATus:OPERation:QENable:NTRansition STATus:OPERation:QENable:NTRansition?

Sets or queries the contents of the Negative Transition Queue Enable Register (NTQER) for the Operation Status Register. When you set an event bit true (1) in the NTQER, the corresponding event the Operation Status Condition Register generates a message in the Status Queue when the event line changes from true to false (1 to 0). The NTQER bits correspond to the Operation Status Register bits. Refer to Table 2–2 on page 2–152 for a definition of the events associated with the Operation Status Register bits. The command STATus:OPERation:QENable:PTRansition provides similar control for positive event transitions. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to zero (0) which disables the reporting of negative event transitions. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:OPERation:QENable:NTRansition <mask>
STATus:OPERation:QENable:NTRansition?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the queue enable register to an illegal value.

Dependencies None

Examples

Command: `STAT:OPER:QEN:NTR #H0001`

Query: `STAT:OPER:QEN:NTR?`

Response: 1

Related Commands

STATus:OPERation?
STATus:OPERation:QENable:PTRansition
STATus:OPERation:CONDition?
STATus:OPERation:NTRansition
STATus:PRESet

STATus:OPERation:QENable:PTRansition STATus:OPERation:QENable:PTRansition?

Sets or queries the contents of the Positive Transition Queue Enable Register (PTQER) for the Operation Status Register. When you set an event bit true (1) in the PTQER, the corresponding event the Operation Status Condition Register generates a message in the Status Queue when the event line changes from false to true (0 to 1). The PTQER bits correspond to the Operation Status Register bits. Refer to Table 2–2 on page 2–152 for a definition of the events associated with the Operation Status Register bits. The command STATus:OPERation:QENable:NTRansition provides similar control for negative event transitions. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to zero (0) which disables reporting positive event transitions. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:OPERation:QENable:PTRansition <mask>
STATus:OPERation:QENable:PTRansition?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the queue enable register to an illegal value.

Dependencies

None

Examples

Command: `STAT:OPER:QEN:PTR #H0001`

Query: `STAT:OPER:QEN:PTR?`

Response: 1

Related Commands

STATus:OPERation?
STATus:OPERation:QENable:NTRansition
STATus:OPERation:CONDition?
STATus:OPERation:NTRansition
STATus:PRESet

STATus:PRESet

Presets the SCPI Enable and Transition registers, and the Status Queue enable registers. The Operation and Questionable Enable Registers are preset to zero (all events disabled). Refer to the register command descriptions for the effects of STATus:PRESet on specific registers. This command does not clear SCPI Status registers or 488.2 enable and event registers. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
STATus:PRESet
```

Parameters

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `STAT:PRES`

Related Commands

STATus:OPERation:ENABle
STATus:OPERation:NTRansition
STATus:OPERation:PTRansition
STATus:OPERation:QENable:NTRansition
STATus:OPERation:QENable:PTRansition
STATus:QUEStionable:ENABle
STATus:QUEStionable:NTRansition
STATus:QUEStionable:PTRansition
STATus:QUEStionable:QENable:NTRansition
STATus:QUEStionable:QENable:PTRansition
STATus:SESR:QENable
*CLS

STATus:QUEStionable? (Query Only)

Returns the contents of the Questionable Status Register as a decimal number. The Questionable Status Register, defined in Table 2–3, identifies operations whose results are questionable. Acquiring a waveform when the waveform analyzer is in need of calibration can generate an event on bit 8. Use the

Operation Status Enable Register to determine which Questionable events can set the Questionable bit (bit 3) in the Status Byte Register. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Reading this register clears it. Unused and reserved bits always return zero.

Table 2–3: The Questionable Status Register

Bit	Decimal Value	Function
0,1,3–7		Not used.
2	4	**Questionable time** indicates the XOFFs or trigger position of the acquired data is questionable.
8	256	**Calibration** indicates that calibration is required due a change of greater than 5° C in ambient temperature since the last calibration.
9	512	**Calculate1** indicates that source data contained a value that was overrange or underrange, making the results of the CALC1 block questionable.
10	1024	**Calculate2** indicates that source data contained a value that was overrange or underrange, making the results of the CALC2 block questionable.
11	2048	**Calculate3** indicates that source data contained a value that was overrange or underrange, making the results of the CALC3 block questionable.
12	4096	**Calculate4** indicates that source data contained a value that was overrange or underrange, making the results of the CALC4 block questionable.
13–15		Not used.

Syntax

```
STATus:QUEStionable[:EVENt]?
```

Parameters

<event>	Query response
Not applicable	<NR1>

Reset Value

Not applicable

Errors and Events

None

Examples Query: `STAT:QUES?`

Response: 1

Related Commands STATus:QUEStionable:CONDition?
STATus:QUEStionable:ENABle

STATus:QUEStionable:CONDition? (Query Only)

Returns the contents of the Questionable Status Condition Register (QSCR). The QSCR bits correspond to the Questionable Status Register bits. The query response from the Questionable Status Condition Register gives the current state of the Questionable event lines prior to the Questionable Transition Registers. The query returns the current setting as a 16 bit number whose bits correspond to the Questionable Status Register bits. Refer to Table 2–3 on page 2–162 for a definition of the events associated with the Questionable Status Register bits. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Note that reading this register does not clear it and unused bits always return zero.

Syntax `STATus:QUEStionable:CONDition?`

Parameters

<condition>	Query response
Not applicable	<NR1>

Reset Value Not applicable

Errors and Events None

Dependencies None

Examples Query: `STAT:QUES:COND?`

Response: 1

Related Commands STATus:QUEStionable?
STATus:QUEStionable:ENABle

STATus:QUEStionable:ENABle
STATus:QUEStionable:ENABle?

Sets or queries the contents of the Questionable Status Enable Register (QSER). The QSER allows you to individually disable any of the Questionable events from setting bit 3 in the Status Byte Register. The query returns the current setting as a decimal number whose bits correspond to the Questionable Status Register bits. Refer to Table 2–3 on page 2–162 for a definition of the events associated with the Questionable Status Register bits. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to zero (0) which disables all Questionable events. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:QUEStionable:ENABle <mask>
STATus:QUEStionable:ENABle?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –222, "Data out of range"
Attempted to set the enable register to an illegal value.

Dependencies

None

Examples

Command: `STAT:QUES:ENAB #H0001`

Query: `STAT:QUES:ENAB?`

Response: 1

Related Commands

STATus:PRESet
STATus:QUEStionable?
STATus:QUEStionable:CONDition?

STATus:QUEStionable:NTRansition STATus:QUEStionable:NTRansition?

Sets or queries the contents of the Questionable Negative Transition Register (QNTR). When you set an event bit true (1) in the QNTR, the QNTR reports an event in the Questionable Status Register when the Questionable Status Condition Register event line changes from true to false (1 to 0). The query returns the current setting as a decimal number whose bits correspond to the Questionable Status Register bits. Refer to Table 2–3 on page 2–162 for a definition of the events associated with the Questionable Status Register bits. The command STATus:QUEStionable:PTRansition provides similar control for positive event transitions. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to zero (0) which disables reporting negative event transitions. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:QUEStionable:NTRansition <mask>
STATus:QUEStionable:NTRansition?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the transition register to an illegal value.

Dependencies

None

Examples

Command: `STAT:QUES:NTR #H0100`
With this command, (decimal 256), the Questionable Negative Transition Register sets the Calibration event bit when the calibration completes.

Query: `STAT:QUES:NTR?`

Response: `256`

Related Commands

STATus:QUEStionable?
STATus:QUEStionable:CONDition?
STATus:QUEStionable:PTRansition
STATus:PRESet

STATus:QUEStionable:PTRansition STATus:QUEStionable:PTRansition?

Sets or queries the contents of the Questionable Positive Transition Register (QPTR). When you set an event bit true (1) in the QPTR, the QPTR reports an event in the Questionable Status Register when the Questionable Status Condition Register event line changes from false to true (0 to 1). The query returns the current setting as a decimal number whose bits correspond to the Questionable Status Register bits. Refer to Table 2–3 on page 2–162 for a definition of the events associated with the Questionable Status Register bits. The command STATus:QUEStionable:NTRansition provides similar control for negative event transitions. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to one (1) which enables reporting positive event transitions. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:QUEStionable:PTRansition <mask>
STATus:QUEStionable:PTRansition?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the transition register to an illegal value.

Dependencies

None

Examples

Command: `STAT:QUES:PTR #H0100`
With this command, (decimal 256), the Questionable Positive Transition Register sets the Calibration event bit when calibration

becomes necessary. All other positive events in the Questionable Condition Register are ignored.

Query: `STAT:QUES:PTR?`

Response: `256`

Related Commands

STATus:QUEStionable?
STATus:QUEStionable:CONDition?
STATus:QUEStionable:NTRansition
STATus:PRESet

STATus:QUEStionable:QENable:NTRansition STATus:QUEStionable:QENable:NTRansition?

Sets or queries the contents of the Negative Transition Queue Enable Register (NTQER) for the Questionable Status Register. When you set an event bit true (1) in the NTQER, the corresponding event the Questionable Status Condition Register generates a message in the Status Queue when the event line changes from true to false (1 to 0). The query returns the current setting as a decimal number whose bits correspond to the Questionable Status Register bits. Refer to Table 2–3 on page 2–162 for a definition of the events associated with the Questionable Status Register bits. The command STATus:QUEStionable:QENable:PTRansition provides similar control for positive event transitions. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to zero (0) which disables reporting negative event transitions. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:QUEStionable:QENable:NTRansition <mask>
STATus:QUEStionable:QENable:NTRansition?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the queue enable register to an illegal value.

Dependencies None

Examples

Command: `STAT:QUES:QEN:NTR #H0100`

Query: `STAT:QUES:QEN:NTR?`

Response: `256`

Related Commands

STATus:QUEStionable?
STATus:QUEStionable:QENable:PTRansition
STATus:QUEStionable:CONDition?
STATus:PRESet

STATus:QUEStionable:QENable:PTRansition
STATus:QUEStionable:QENable:PTRansition?

Sets or queries the contents of the Positive Transition Queue Enable Register (PTQER) for the Questionable Status Register. When you set an event bit true (1) in the PTQER, the corresponding event the Questionable Status Condition Register generates a message in the Status Queue when the event line changes from false to true (0 to 1). The query returns the current setting as a decimal number whose bits correspond to the Questionable Status Register bits. Refer to Table 2–3 on page 2–162 for a definition of the events associated with the Questionable Status Register bits. The command STATus:QUEStionable:QENable:NTRansition provides similar control for negative event transitions. For an overview of the status and event reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

STATus:PRESet sets all register bits to zero (0) which disables reporting positive event transitions. Setting unused enable bits does not generate an error, they are ignored.

Syntax

```
STATus:QUEStionable:QENable:PTRansition <mask>
STATus:QUEStionable:QENable:PTRansition?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	<NR1>

Reset Value Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the queue enable register to an illegal value.

Dependencies

None

Examples

Command: `STAT:QUES:QEN:PTR #H0100`

Query: `STAT:QUES:QEN:PTR?`

Response: `256`

Related Commands

STATus:QUEStionable?
STATus:QUEStionable:QENable:NTRansition
STATus:QUEStionable:CONDition?
STATus:PRESet

STATus:SESR:QENable
STATus:SESR:QENable?

Sets or returns the contents of the Event Status Enable Register (ESER). When you set an event bit true (1) in the ESER, the corresponding event the Standard Event Status Register generates a message in the Status Queue when the event line changes from false to true (0 to 1).

STATus:PRESet sets the register bits to hexadecimal 3C.

Syntax

```
STATus:SESR:QENable <mask>
STATus:SESR:QENable?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFFFF	None

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the queue enable register to an illegal value.

Dependencies

None

Examples

Command: `STAT:SESR:QEN #H7C`

Query: `STAT:SESR:QEN?`

Response: `124`
This response is the decimal equivalent of hexadecimal 7C.

Related Commands

*ESE
*ESR
STATus:PRESet

SWEep Subsystem

This section describes the commands in the [SENSe:]SWEep subsystem. See Figure 2–21 and Figure 2–22. These commands control the acquisition timebase for all VOLTage[n] acquisitions.

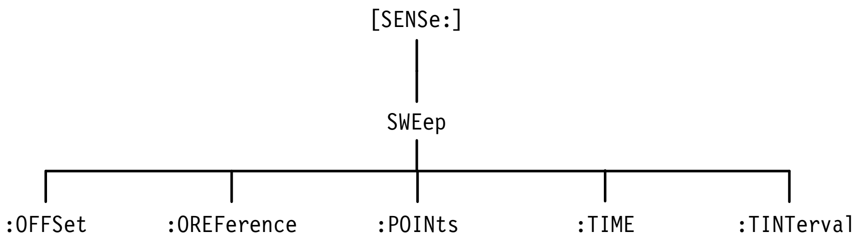


Figure 2–21: SWEep subsystem hierarchy

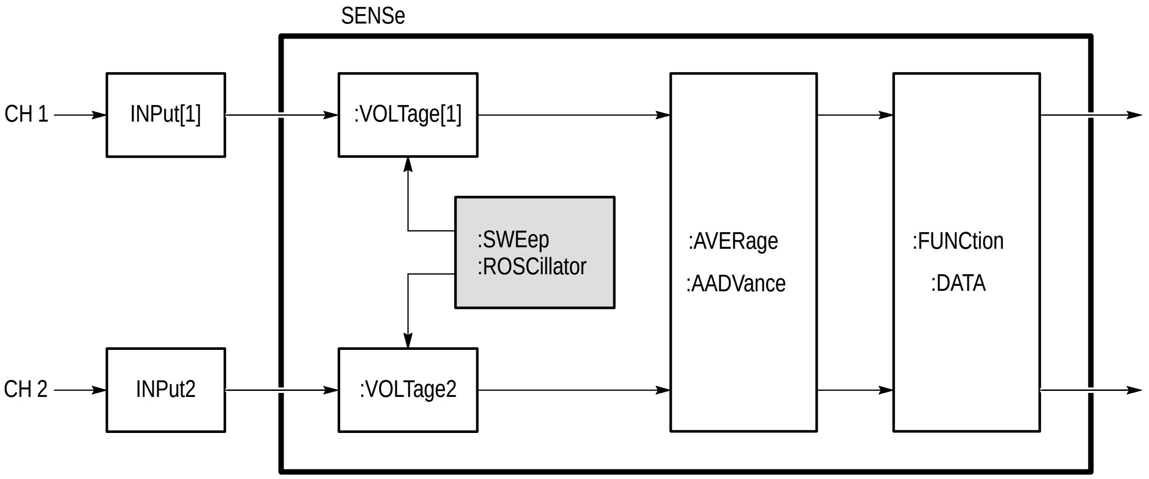


Figure 2–22: SWEep subsystem functional model

SWEep:OFFSet:POINts
SWEep:OFFSet:POINts?

Sets or queries the position of the waveform record relative to the trigger point. SWEep:OFFSet:POINts defines the number of data points between the offset reference point and the trigger point. You set the offset reference with SWEep:OREFerence:LOCation. All channels use the same sweep offset.

You can think of the offset reference point as a “handle” that you place on a specific data point in the waveform record. You position the “handle” to move the waveform record relative to the trigger point. The command OFFSet:POINts specifies how far and in what direction to move the “handle.” Setting OFFSet:POINts to a negative value positions the offset reference (the handle) before the trigger point. A positive value positions the offset reference after the trigger point.

For example, to set the trigger point in the middle of the waveform record, you would first set the offset reference point to the first record point (SWE:OREF:LOC 0.0). Then you would set SWE:OFFS:POIN to minus one half the record length (set with SWEep:POINTs). If the record length is 1024, then to specify the half way point you would use SWE:OFFS:POIN –512.

The range for SWEep:OFFSet:POINts depends on the record length and the location of the offset reference point within the waveform record. Note that the trigger point must always be within the waveform record. You can set offset as a time interval with the command OFFSet:TIME.

The following equation defines the first point of the waveform record:

$$PT1_{time} = (\text{SWEep:OFFSet:POINts} \times \text{SWEep:TINTerval}) - (\text{SWEep:OREFerence:LOCation} \times \text{SWEep:TIME})$$

For more information, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
[SENSe:]SWEep:OFFSet:POINts <offset_points>
[SENSe:]SWEep:OFFSet:POINts?
```

Parameters

<offset_points>	Query response
<NRf>	<NR1>
MINimum	(SWEep:OREFerence:LOCation × SWEep:POINts) – SWEep:POINts
MAXimum	(SWEep:OREFerence:LOCation × SWEep:POINts)

Reset Value

0.0E+0

Errors and Events

Execution Error –222, "Data out of range"
Attempted to set offset points to an illegal value.

Dependencies

Changing SWEep:OFFSet:POINts also changes SWEep:OFFSet:TIME as they both control the parameter SWEep:OFFSet.

Examples

Command: `SWE:OFFS:POIN -100`

Query: `SWE:OFF:POIN?`

Response: `-100`

Related Commands

SWEep:OREFerence:LOCation
SWEep:OFFSet:TIME
SWEep:POINts
SWEep:TINTerval

SWEep:OFFSet:TIME
SWEep:OFFSet:TIME?

Sets or queries the offset between the offset reference point and the trigger point. You define OFFSet:TIME as a specific period. You assign a certain data point in the record to be the offset reference point (SWEep:OREFerence:LOCation). A negative value for offset positions the offset reference point before the trigger point. All channels use the same sweep offset.

The range for SWEep:OFFSet:TIME depends on the record length and the location of the offset reference point within the waveform record. Because the trigger point must always be in the waveform record, the first point of the record can not be placed more than one full record length before the trigger point. You can set offset as a number of data points with the command SWEep:OFFSet:POINts.

The following equation defines the first point of the waveform record:

$PT1_{time}$ = SWEep:OFFSet:TIME –
(SWEep:OREFerence:LOCation * SWEep:TIME)

For more information on setting offset, refer to the discussion of *SWEep:OFFSet:POINts* on page 2–172 and the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
[SENSe:]SWEep:OFFSet:TIME <offset_time>
[SENSe:]SWEep:OFFSet:TIME?
```

Parameters

<offset_time>[1]	Query response
<NRf>	<NR3>
MINimum	(SWEep:OREFerence:LOCation × SWEep:TIME) – (SWEep:TIME)
MAXimum	(SWEep:OREFerence:LOCation × SWEep:TIME)

[1] The default multiplier for <offset_time> is S for seconds. You can also use the multipliers MS for milliseconds, US for microseconds, NS for nanoseconds, and PS for picoseconds.

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set offset time to an illegal value.

Dependencies

Changing SWEep:OFFSet:TIME also changes SWEep:OFFSet:POINts as they both control the parameter SWEep:OFFSet.

Examples

Command: `SWE:OFFSet:TIME -100E-6`

Query: `SWE:OFFSet:TIME?`

Response: `-100.0E-6`

Related Commands

SWEep:OREFerence:LOCation
SWEep:OFFSet:POINts
SWEep:TINTerval
SWEep:POINts

SWEep:OREFerence:LOCation SWEep:OREFerence:LOCation?

Sets or queries the location of the offset reference point in a waveform record. When :OFFSet:TIME and :OFFSet:POINts are at zero, the reference point and trigger point are at the same data point. For example, if you set :OREFerence:LOCation to 0.5 (or 50%) and :OFFSet:TIME to zero, then the reference point is at the middle of the waveform record as is the trigger point. Therefore, half of the acquired data points are acquired before the trigger point and the remainder occur after the trigger point.

An offset reference value of 0.0 selects the first point of the record; a value of 0.5 selects the mid point; and a value of 1.0 selects the last point. All concurrent acquisitions have the same offset reference point.

You can use either of the commands SWEep:OFFSet:TIME or SWEep:OFFSet:POINts to position the waveform record relative to the trigger point. To configure an acquisition in terms of percent of pre-trigger data points, set SWEep:OFFSet:TIME to 0.0 and then set SWEep:OREFerence:LOCation to the desired value (for example, 0.5 for 50% pre-trigger points). To configure an acquisition in terms of time (or points) relative to the trigger point, set SWEep:OREFerence:LOCation to 0.0 and then set SWEep:OFFSet:TIME (or SWEep:OFFSet:POINts) to position the record. By using :OFFSet in this way, you can acquire all data points before the trigger or all data points after the trigger point.

For more information, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
[SENSe:]SWEep:OREFerence:LOCation <oref_location>
[SENSe:]SWEep:OREFerence:LOCation?
```

Parameters

<oref_location>[1]	Query response
No Units: <NRf> 0.0 ≤ N ≤ 1.0 MINimum MAXimum	 <NR2> 0.0 1.0
Units of Percent (PCT): <NRf> 0 ≤ N ≤ 100 MINimum (0 PCT) MAXimum (100 PCT)	 <NR2> 0 1.0

[1] The default has no units. However, you can set the reference location as a percentage by including the unit multiplier PCT as in 50 PCT.

Reset Value

0.0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set offset reference location to an illegal value.

Dependencies

None

Examples

Command: `SWE:OREF:LOC 0.5`

Query: `SWE:OREF:LOC?`

Response: `0.5`

Related Commands

SWEep:OFFSet:TIME
SWEep:OFFSet:POINts

SWEep:POINts
SWEep:POINts?

Sets or queries the number of data points in a waveform record. The number of data points in a waveform record is its record length. The minimum record length is 256 data points and the maximum is 15,000 real time (RT) acquisition and 30,000 for extended real time (ERT) acquisition. All active channels share the record length setting.

The maximum record length for calculation or measurement functions (using a CALCulate block) or for auto-advance acquisition mode is 30,000.

Syntax

```
[SENSe:]SWEep:POINts <record_length>
[SENSe:]SWEep:POINts?
```

Parameters

<record_length> (Number of data points)	Query response
<NRf>	<NR1>
256	256
512	512
1024	1024
2048	2048
4096	4096
8192	8192
15000	15000
30000 (ERT mode only)	30000
MINimum	256
MAXimum RT mode	15000
MAXimum ERT mode	30000

Reset Value

1024

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set record length:
to an illegal value or
to a value > 15,000 while in real time acquisition or
to a value > 30,000 while in extended real time acquisition.

Dependencies

Changing SWEep:POINts changes SWEep:TIME if SWEep:TINTerval remains constant. The relationship is
SWEep:TINTerval = SWEep:TIME / SWEep:POINts.

Examples

Command: `SWE:POIN 1000`

Query: `SWE:POIN?`

Response: `1024`

Related Commands

SWEep:TINTerval
SWEep:TIME

SWEep:TIME
SWEep:TIME?

Sets or queries the time span or duration of the waveform record. All channels share the same time span. Set SWEep:POINts for the desired record length before setting the record duration with SWEep:TIME. Issuing these two commands sets the sample interval (SWEep:TINTerval) to the nearest legal value.

When using SWEep:POINts and SWEep:TIME to configure the acquisition, be careful to program SWEep:POINts before SWEep:TIME, because SWEep:POINts may modify SWEep:TIME.

SWEep:TIME is not returned in the response to the query *LRN?, since it can be derived as follows: SWEepTIME = SWEep:POINts * SWEep:TINTerval

Syntax

```
[SENSe:]SWEep:TIME <time_span>
[SENSe:]SWEep:TIME?
```

Parameters

<time_span> [1]	Query response
<NRf> 51.2E–9 ≤ N ≤ 6000 (seconds)	<NR3>
MINimum	256 × sample interval
MAXimum	6000.0E+0

[1] **The default multiplier for <time_span> is S for seconds. You can also use the multipliers MS for milliseconds, US for microseconds, NS for nanoseconds, and PS for picoseconds.**

Reset Value

1.024E–6

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set time span to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to set time span to 51.2E–9 when the instrument is a TVS621A or TVS641A.

Execution Error –241, “Hardware missing”
Attempted to set time span to 51.2E–9 when the instrument is a TVS621A, TVS623A, TVS643A or TVS641A or to 102E–9 when the instrument is a TVS621A or TVS641A.

Dependencies

Changing SWEep:TIME changes SWEep:TINTerval if SWEep:POINts remains constant. The relation is SWEep:TINTerval = SWEep:TIME / SWEep:POINts.

Examples

Command: `SWE:TIME 1E-3`

Query: `SWE:TIME?`

Response: `1.024E-3`

Related Commands

SWEep:POINts
SWEep:TINTerval

SWEep:TINTerval
SWEep:TINTerval?

Sets or queries the time interval between acquired data points. The reciprocal of the time interval gives the effective sample rate. All channels are acquired with the same time interval.

The point at which the instrument transitions from real time (RT) acquisition to extended real time (ERT) acquisition mode is independent of instrument configuration or settings.

RT < 100 ns ≥ ERT

Syntax

```
[SENSe:]SWEep:TINTerval <time_interval>
[SENSe:]SWEep:TINTerval?
```

Parameters

<time_interval> (seconds)[1]	Query response
<NRf> 200E–12 (TVS625A, TVS645A only) 400E–12 (TVS625A, TVS645A only) 1E–9 2E–9 4E–9 10E–9 20E–9 40E–9 100E–9 . . . 200E–3	<NR3> 200.0E–12 400.0E–12 1.0E–9 2.0E–9 4.0E–9 10.0E–9 20.0E–9 40.0E–9 100.0E–9 . . . 200.0E–3
MINimum	depends on model
MAXimum	200.0E–3

[1] The default multiplier for <time_interval> is S for seconds. You can also use the multipliers MS for milliseconds, US for microseconds, NS for nanoseconds, and PS for picoseconds.

Reset Value

1.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set time interval to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to set time interval to 200E–12 or 400E–12 when the instrument is a TVS621 or TVS641.

Execution Error –241, "Hardware missing"
Attempted to set time interval to 200E–12 when the instrument is a TVS621A, TVS641A; or to set time interval to 400E-12 when the instrument is a TVS621A or TVS641A.

Dependencies

Changing SWEep:TINTerval changes SWEep:TIME. The relation is SWEep:TINTerval = SWEep:TIME / SWEep:POINts.

Changing SWEep:TIMTerval to < 100 ns changes SWEep:POINts to 15,000 if previously set to 30,000. The acquisition mode changes from extended real time to real time mode below 100 ns/sample.

Examples

Command: `SWE:TINT 1E-6`

Query: `SWE:TINT?`

Response: `1.0E-6`

Related Commands

SWEep:POINts
SWEep:TIME

SYSTem Subsystem

This section describes each command and query in the SYSTem subsystem. See Figure 2–23. These commands program utility functions and return version information about the waveform analyzer.

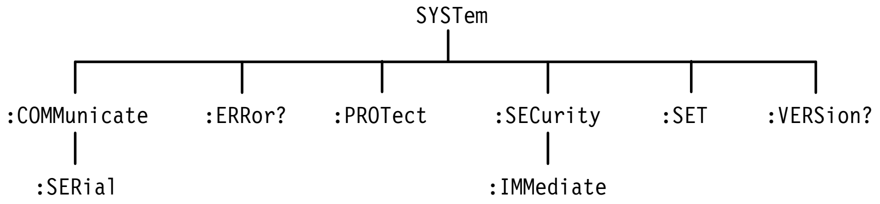


Figure 2–23: SYSTem subsystem hierarchy

SYSTem:AUToset:SWEep

TVS600A Models Only

Sets SWEep:TINTerval to the nearest value such that an acquired record contains 2.5 cycles of the incoming signal on the specified channel.

A main-to-delay trigger timer is used to measure the period of the requested channel. The main and delay triggers will be temporarily assigned to the requested channel and levels set to 50% of the signal while this command is executing. The trigger search will timeout after approximately 20ms. This implies that the incoming signal should be a repetitive 50Hz signal or faster.

If no triggers are found then the sweep rate will be unchanged.

It is the responsibility of the host application to preset all related settings such as bandwidth filters, couplings, vertical settings, and so on.

After autoset completes, the host application can query SWEep:TINTerval to find out what was done.

When part of a global autoset, SYSTem:AUToset:VOLTage should be sent first.

Syntax

```
SYSTem:AUToset:SWEep <channel>
```

Parameters

<channel>	Query response
CHAN[1] CHAN2 CHAN3 CHAN4	Not applicable

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

Command Error –141, “Invalid character data”
Attempted to autoset IMMediate or any other illegal value.

Execution Error –221, “Settings conflict”
Attempted to autoset a channel that is not turned on.

Execution Error –241, “Hardware missing”
Attempted to autoset INTernal3 or INTernal4 when the instrument is configured with only two channels.

Dependencies See SWEep:TINTerval side effects.

Examples Command: `SYST:AUT:SWE CHAN1`

Related Commands SWEep:TINTerval
SYSTem:AUToset:VOLTage
SYSTem:AUToset:TRIGger

SYSTem:AUToset:TRIGger

TVS600A Models Only

This command sets the TRIGger[:A]:SOURe to <channel> and TRIGger:LEVel to 50% of the peak to peak value of the trigger signal.

A binary search algorithm will be used with the trigger comparator to find the peaks of the trigger signal. The trigger search will timeout after approximately 20ms. This implies that the incoming signal should be a repetitive 50Hz signal or faster.

If no triggers are found then the trigger level will be unchanged.

It is the responsibility of the host application to preset all related settings such as bandwidth filters, couplings, vertical settings, and so on.

After autoset completes, the host application can query TRIGger:LEVel to find out what was done.

When part of a global autoset, SYSTem:AUToset:VOLTage should be performed first.

Syntax `SYSTem:AUToset:TRIGger <channel>`

Parameters

<channel>	Query response
CHAN[1] CHAN2 CHAN3 CHAN4	Not applicable

Suffix Units and Multipliers None

Reset Value Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `SYSTem:AUT:TRIG CHAN`

Related Commands

TRIGger:LEVel
SYSTem:AUToset:VOLTage
SYSTem:AUToset:SWEep

SYSTem:AUToset:VOLTage

TVS600A Models Only

Sets VOLTage:RANGe:UPPer and VOLTage:RANGe:LOWer so that the incoming signal fills the center 90% of the peak to peak range.

Multiple acquisitions of an arbitrary period of 20ms will be acquired with peak detect circuits enabled to find the peaks of the signal. This implies that the incoming signal should be a repetitive 50Hz signal or faster.

It is the responsibility of the host application to preset all related settings such as bandwidth filters, couplings, vertical settings, and so on.

After autoset completes, the host application can query VOLTage:RANGe:UPPer and LOWer or VOLTage:RANGe PTPeak and OFFSet to find out what was done.

Syntax

```
SYSTem:AUToset:VOLTage <channel>
```

Parameters

<channel>	Query response
CHAN[1] CHAN2 CHAN3 CHAN4	Not applicable

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

Command Error –141, “Invalid character data”
Attempted to autoset IMMediate or any other illegal value.

Execution Error –221, “Settings conflict”
Attempted to autoset a channel that is not turned on.

Execution Error –241, “Hardware missing”
Attempted to autoset INTernal3 or INTernal4 when the instrument is configured with only two channels.

Dependencies

Trigger level may be affected.

Examples

Command: `SYST:AUT:VOLT CHAN1`

Related Commands

VOLTage:RANGe:UPPer
VOLTage:RANGe:LOWer
SYSTem:AUToset:SWEep
SYSTem:AUToset:TRIGger

SYSTem:BDATe?
SYSTem:BDATe

TVS600A Models Only

Sets and queries the battery installation date for the nonvolatile memory in the instruments.

The date loaded is not evaluated for accuracy by the instrument. However the year should be the complete four digit year ie. 1997 and the month should be 1 through 12 for January through December and the day should be 1 through 31 for the day of the month.

Syntax

```
SYSTem:BDATe?
SYSTem:BDATe <year>,<month>,<day>
```

Parameters

<year> <month> <day>	Query response
<NRf>, <NRf>, <NRf>	<NR1>,<NR1>,<NR1>

Suffix Units and Multipliers None

Reset Value Not applicable

Errors and Events Execution Error –203, “Command protected”
Attempted to set battery date when SYSTem:PROTect is ON.

Dependencies None

Examples

Command: `SYST:BDAT 1997,7,4`

Query: `SYST:BDAT?`

Response: `1997,7,4`

Related Commands SYSTem:PROTect

SYSTem:CDATe?
SYSTem:CDATe

TVS600A Models Only

Sets and queries the factory/service calibration date of the instruments.

The date loaded is not evaluated for accuracy by the instrument. However the year should be the complete four digit year (for example, 1997), the month should be 1 through 12 for January through December, the day should be 1 through 31 for the day of the month.

Syntax

```
SYSTem:CDATe?
SYSTem:CDATe <year>,<month>,<day>
```

Parameters

<year> <month> <day>	Query response
<NRf>, <NRf>, <NRf>	<NR1>,<NR1>,<NR1>

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

Execution Error –203, “Command protected”
Attempted to set battery date when SYSTem:PROTect is ON.

Dependencies

None

Examples

Command: `SYST:CDAT 1997,7,4`

Query: `SYST:CDAT?`

Response: `1997,7,4`

Related Commands

SYSTem:PROTect

SYSTem:COMMunicate:SERial:BAUD SYSTem:COMMunicate:SERial:BAUD?

Sets or queries the baud rate of the front panel RS-232 port. The baud rate is the data transfer rate in bits per second for a serial transfer. Both transmit and receive baud rates are set by this command. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

When you set :SERial:BAUD, the new value is stored in nonvolatile RAM and is recalled at power up. However, the baud rate is not saved with stored settings (*LRN) and it is not reset to a default value by *RST.

Syntax

```
SYSTem:COMMunicate:SERial:BAUD <baud_rate>
SYSTem:COMMunicate:SERial:BAUD?
```

Parameters

<baud_rate>	Query response
<NRf>	<NR1>
300	300
600	600
1200	1200
2400	2400
4800	4800
9600	9600
19200	19200
38400[1]	38400[1]
57600[1]	57600[1]
MINimum	300
MAXimum	19200 (57600[1])

[1] **TVS600A only**

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the baud rate to an illegal value.

Dependencies

None

Examples

Command: `SYST:COMM:SER:BAUD 4800`

Query: `SYST:COMM:SER:BAUD?`

Response: `4800`

Related Commands

SYSTem:COMMunicate:SERial:PARity
SYSTem:COMMunicate:SERial:SBITs

SYSTem:COMMunicate:SERial:CONTrol:DCD
SYSTem:COMMunicate:SERial:CONTrol:DCD?

Sets or queries whether the instrument is sensitive to the front panel RS-232 DCD line. The setting ON specifies that the port ignores data unless the DCD line is high. This mode is useful when you have a modem connected. The setting OFF means the DCD line is ignored. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

When you set :SERial:CONTrol:DCD, the new value is stored in nonvolatile RAM and is recalled at power up. However, the control mode is not saved with stored settings (*LRN) and it is not reset to a default value by *RST.

Syntax

```
SYSTem:COMMunicate:SERial:CONTrol:DCD <boolean>
SYSTem:COMMunicate:SERial:CONTrol:DCD?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `SYST:COMM:SER:CONT:DCD ON`

Query: `SYST:COMM:SER:CONT:DCD?`

Response: 1

Related Commands

SYSTem:COMMunicate:SERial:CONTrol:RTS
SYSTem:COMMunicate:SERial:PACE

SYSTem:COMMunicate:SERial:CONTrol:RTS SYSTem:COMMunicate:SERial:CONTrol:RTS?

Sets or queries the operation of the front panel RS-232 RTS and CTS lines. The setting OFF sets the outgoing RTS line low and ignores the incoming CTS line. The setting ON sets the outgoing RTS line high and ignores the incoming CTS line. The setting IBFull or RFR enables the two lines for hardware flow control. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

The waveform analyzer does not support DTR/DSR flow control. The instrument holds the DTR line high and ignores the DSR line.

When you set :SERial:CONTrol:RTS, the new value is stored in nonvolatile RAM and is recalled at power up. However, the control mode is not saved with stored settings (*LRN) and it is not reset to a default value by *RST.

Syntax

```
SYSTem:COMMunicate:SERial:CONTrol:RTS <flow_control>
SYSTem:COMMunicate:SERial:CONTrol:RTS?
```

Parameters

<flow_control>	Query response
OFF	OFF
ON	ON
IBFull | RFR	IBF

Reset Value

Not applicable

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set the hardware flow control to an illegal value.

Dependencies

None

Examples

Command: `SYST:COMM:SER:CONT:RTS IBF`

Query: `SYST:COMM:SER:CONT:RTS?`

Response: `IBF`

Related Commands

SYSTem:COMMunicate:SERial:CONTrol:DCD
SYSTem:COMMunicate:SERial:PACE

SYSTem:COMMunicate:SERial:ECHO SYSTem:COMMunicate:SERial:ECHO?

Sets or queries whether incoming characters are echoed back to the front panel RS-232 port. This mode is useful when you are typing commands at a connected terminal because you can see what characters the instrument received. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

When you set :SERial:ECHO, the new value is stored in nonvolatile RAM and is recalled at power up. However, the echo mode is not saved with stored settings (*LRN) and it is not reset to a default value by *RST.

Syntax

```
SYSTem:COMMunicate:SERial:ECHO <boolean>
SYSTem:COMMunicate:SERial:ECHO?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `SYST:COMM:SER:ECHO OFF`

Query: `SYST:COMM:SER:ECHO?`

Response: `0`

Related Commands

SYSTem:COMMunicate:SERial:PRESet:RAW
SYSTem:COMMunicate:SERial:PRESet:TERMinal

SYSTem:COMMunicate:SERial:ERESponse SYSTem:COMMunicate:SERial:ERESponse?

Sets or queries whether error messages are automatically returned to the front panel RS-232 port. When OFF, error messages are stored in the RS-232 Error

Queue. The OFF mode is appropriate when you have a computer connected to the RS-232 connector. When ON, messages are immediately displayed. The ON mode is most appropriate when a simple display terminal is connected to the front panel RS-232. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

When you set :SERial:ERESponse, the new value is stored in nonvolatile RAM and is recalled at power up. However, the error response mode is not saved with stored settings (*LRN) and it is not reset to a default value by *RST.

Syntax

```
SYSTem:COMMunicate:SERial:ERESponse <boolean>
SYSTem:COMMunicate:SERial:ERESponse?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value Not applicable

Errors and Events None

Dependencies None

Examples

Command: `SYST:COMM:SER:ERES OFF`

Query: `SYST:COMM:SER:ERES?`

Response: `0`

Related Commands SYSTem:COMMunicate:SERial:PRESet:RAW
SYSTem:COMMunicate:SERial:PRESet:TERMinal

SYSTem:COMMunicate:SERial:LBUFfer
SYSTem:COMMunicate:SERial:LBUFfer?

Sets or queries the state of the character buffer for the front panel RS-232 port. When ON, all input characters are queued until a newline (^J) or return (^M) character is received; then the buffer contents are released to the command parser. The following line editing commands let you modify the input before entering a newline or return character:

- ^H (backspace),
- ^U (line-delete)
- ^D (end-of-file).

When OFF, characters are immediately passed to the parser, line editing is not possible, and you must enter a newline (^J) character to terminate a command line. When transferring binary data, always set the line buffer to OFF. When buffering is on, return characters (^M) are converted to newlines (^J) and outgoing newlines (^J) are converted to return-newlines (^M^J). The conversion of termination characters causes significant problems when transferring binary data.

When you set :SERial:LBUFfer, the new value is stored in nonvolatile RAM and is recalled at power up. However, the buffer mode is not saved with stored settings (*LRN) and it is not reset to a default value by *RST. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

Syntax

```
SYSTem:COMMunicate:SERial:LBUFfer <boolean>
SYSTem:COMMunicate:SERial:LBUFfer?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `SYST:COMM:SER:LBUF OFF`

Query: `SYST:COMM:SER:LBUF?`

Response: 0

Related Commands

SYSTem:COMMunicate:SERial:PRESet:RAW
SYSTem:COMMunicate:SERial:PRESet:TERMinal

SYSTem:COMMunicate:SERial:PACE SYSTem:COMMunicate:SERial:PACE?

Sets or queries whether software flow control (XON/XOFF) is enabled for the front panel RS-232 port. Both transmit and receive values are set by this command. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

When you set :SERial:PACE, the new value is stored in nonvolatile RAM and is recalled at power up. However, the flow control mode is not saved with stored settings (*LRN) and it is not reset to a default value by *RST.

You should not use software flow control when transferring binary data.

Syntax

```
SYSTem:COMMunicate:SERial:PACE <flow_control>
SYSTem:COMMunicate:SERial:PACE?
```

Parameters

<flow_control>	Query response
NONE XON	NONE XON

Reset Value

Not applicable

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set the software flow control to an illegal value.

Dependencies

None

Examples

Command: `SYST:COMM:SER:PACE XON`

Query: `SYST:COMM:SER:PACE?`

Response: `XON`

Related Commands

SYSTem:COMMunicate:SERial:CONTrol:DCD
SYSTem:COMMunicate:SERial:CONTrol:RTS
SYSTem:COMMunicate:SERial:PRESet:RAW
SYSTem:COMMunicate:SERial:PRESet:TERMinal

SYSTem:COMMunicate:SERial:PARity SYSTem:COMMunicate:SERial:PARity?

Sets or queries the type of parity for the front panel RS-232 port. Parity provides a minimum level of data security by appending one data bit to each transmitted character to achieve an even or odd number of “1” digits. Both transmit and receive values are set by this command. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

When you set :SERial:PARity, the new value is stored in nonvolatile RAM and is recalled at power up. However, the parity mode is not saved with stored settings (*LRN) and it is not reset to a default value by *RST.

Syntax

```
SYSTem:COMMunicate:SERial:PARity[:TYPE] <type>
SYSTem:COMMunicate:SERial:PARity[:TYPE]?
```

Parameters

<type> (Select parity type)	Query response
EVEN	EVEN
NONE	NONE
ODD	ODD

Reset Value

Not applicable

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set the parity to an illegal value.

Dependencies

None

Examples

Command: `SYST:COMM:SER:PAR EVEN`

Query: `SYST:COMM:SER:PAR?`

Response: `EVEN`

Related Commands

SYSTem:COMMunicate:SERial:BAUD
SYSTem:COMMunicate:SERial:SBITs

SYSTem:COMMunicate:SERial:PRESet

These commands provide three :SERial settings that configure the RS-232 port parameters for typical transfer modes. The default node sets all RS-232 settings to a known state. :RAW and :TERMinal change only a subset of the RS-232 parameters. The :RAW preset is most appropriate when you connect a computer to the RS-232 port. TERMinal should be used when you connect a display terminal.

The command “T” followed by a newline (^J) will configure the RS-232 port for terminal operation when it is currently in RAW mode. There are no associated queries.

The Table 2–4 specifies the preset values for each PRESet command.

Table 2–4: Effects of :PRESet on Serial Port Parameters

SERial Port Parameter	[:ALL]	:RAW	:TERMinal
BAUD	9600	NC[1]	NC[1]
CONTrol:DCD	OFF	NC[1]	NC[1]
CONTrol:RTS	ON	NC[1]	NC[1]
ECHO	OFF	OFF	ON
ERESponse	OFF	OFF	ON
LBUFfer	OFF	OFF	ON
PACE	NONE	NONE	XON
PARity	NONE	NC[1]	NC[1]
SBITs	1	NC[1]	NC[1]

[1] **The entry NC means the value is not changed.**

Syntax

```
SYSTem:COMMunicate:SERial:PRESet[:ALL]
SYSTem:COMMunicate:SERial:PRESet:RAW
SYSTem:COMMunicate:SERial:PRESet:TERMinal
```

Parameters

Not applicable	Query response
See Syntax	Not applicable

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `SYST:COMM:SER:PRES:TERM`

Related Commands

SYSTem:COMMunicate:SERial Commands

SYSTem:COMMunicate:SERial:SBITs SYSTem:COMMunicate:SERial:SBITs?

Sets or queries the number of stop bits sent with each character transmitted on the front panel RS-232 port. Both transmit and receive values are set by this command. Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for the factory setting.

When you set :SERial:SBITs, the new value is stored in nonvolatile RAM and is recalled at power up. However, the number of stop bits is not saved with stored settings (*LRN) and it is not reset to a default value by *RST.

Syntax

```
SYSTem:COMMunicate:SERial:SBITs <stop_bits>
SYSTem:COMMunicate:SERial:SBITs?
```

Parameters

<stop_bits>	Query response
<NRf> 1 or MINimum 2 or MAXimum	<NR1> 1 2

Reset Value

Not applicable

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set the number of stop bits to an illegal value.

Dependencies

None

Examples

Command: `SYST:COMM:SER:SBIT 2`

Query: `SYST:COMM:SER:SBIT?`

Response: `2`

Related Commands

SYSTem:COMMunicate:SERial:BAUD
SYSTem:COMMunicate:SERial:PARity

SYSTem:ERRor? (Query Only)

This query returns the next entry from the waveform analyzer Status Queue. The event message contains the event number and a text description of the event. Reading an event removes it from the queue. Errors and events are cleared from the Status Queue at power up, upon receipt of the command *CLS, and upon reading the last item from the event queue.

Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a detailed description of the Status Queue and a complete list of system events.

Syntax

```
SYSTem:ERRor[:NEXT]?
```

Parameters

<event>	Query response
Not applicable	<NR1>, <string>

Reset Value

Not applicable

Errors and Events

Device Specific Error –350, "Queue Overflow"
The error/event queue overflowed due to the execution errors of other commands. The SYST:ERR? query cannot generate an error.

Dependencies

None

Examples

Query: `SYST:ERR?`

Response: `–221, "Settings conflict"`

Related Commands

STATus:OPERation:QENable:NTRansition
STATus:OPERation:QENable:PTRansition
STATus:QUEStionable:QENable:NTRansition
STATus:QUEStionable:QENable:PTRansition
STATus:PRESet
STATus:SESR:QENable
SYSTem:ERRor:ALL?
*CLS

SYSTem:ERRor:ALL? (Query Only)

Returns the list of all events stored in the waveform analyzer Status Queue. Each event message contains the event number and a text description of the event. Events are separated by commas. Errors and events are cleared from the Status Queue at power up, upon receipt of the command *CLS, and upon reading the last item from the event queue.

Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a detailed description of the Status Queue and a complete list of system events.

Syntax

```
SYSTem:ERRor:ALL?
```

Parameters

<event>	Query response
Not applicable	<NR1>, <string>

Reset Value

Not applicable

Errors and Events

0, “No error”
Attempted to read an event when the queue is empty.

Device Specific Error –350, “Queue Overflow”
The Status Queue has overflowed or run out of memory.

Dependencies

None

Examples

Query: `SYST:QUE:ALL?`

Response: `–221, "Settings conflict",–224,"Illegal parameter value"`

Related Commands

SYSTem:ERRor?

SYSTem:ERRor:CODE? (Query Only)

Returns the next event codes stored in the waveform analyzer Status Queue. The returned error code does not include the associated event message. Errors and events are cleared from the Status Queue at power up, upon receipt of the command *CLS, and upon reading the last item from the event queue. To return all error codes use SYSTem:ERRor:CODE:ALL?. Use SYSTem:ERRor? to return an error code and its event message.

Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a detailed description of the Status Queue and a complete list of system events.

Syntax

```
SYSTem:ERRor:CODE[:NEXT]?
```

Parameters

<code>	Query response
Not applicable	<NR1>

Reset Value

Not applicable

Errors and Events

0, “No error”
Attempted to read an event when the queue is empty.

Device Specific Error –350, “Queue Overflow”
The Status Queue has overflowed or run out of memory.

Dependencies

None

Examples

Query: `SYST:ERR:CODE?`

Response: `–221`

Related Commands

SYSTem:ERRor?
SYSTem:ERRor:CODE:ALL?

SYSTem:ERRor:CODE:ALL? (Query Only)

Returns the list of all event codes stored in the waveform analyzer Status Queue. The returned error codes do not include their associated event messages. Event codes are separated by commas. Use SYSTem:ERRor:CODE? to return one error code at a time. Use SYSTem:ERRor:ALL? to return all error codes and event messages.

Refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for a detailed description of the Status Queue and a complete list of system events.

Syntax

```
SYSTem:ERRor:CODE:ALL?
```

Parameters

<code>	Query response
Not applicable	<NR1>

Reset Value

Not applicable

Errors and Events

0, “No error”
Attempted to read an event when the queue is empty.

Device Specific Error –350, “Queue Overflow”
The Status Queue has overflowed or run out of memory.

Dependencies

None

Examples

Query: `SYST:ERR:CODE:ALL?`

Response: `–221,–224`

Related Commands

SYSTem:ERRor?
SYSTem:ERRor:CODE

SYSTem:ERRor:COUNt? (Query Only)

Returns the number of unread events in the Status Queue. The Status Queue is not modified by this command. Use SYSTem:ERRor? to return the errors and events from the Status Queue. Errors and events are cleared from the Status Queue at power up, upon receipt of the command *CLS, and upon reading the last item from the event queue.

Syntax

```
SYSTem:ERRor:COUNt?
```

Parameters

<count>	Query response
Not applicable	<NR1>

Reset Value Not applicable

Errors and Events None

Dependencies None

Examples

Query: `SYST:ERR:COUN?`

Response: `3`

Related Commands

SYSTem:ERRor:ALL?
SYSTem:ERRor:CODE:ALL?

SYSTem:PROTect
SYSTem:PROTect?

Sets or queries whether protection for a group of sensitive instrument commands is enabled. When protection is enabled (set ON, the default), the following settings cannot be changed without first setting SYSTem:PROTect to OFF:

- INPut:PROTection:STATe
- SYSTem:SECurity:IMMediate
- TEST:LOG:CLEar
- *PUD

SYSTem:PROTect is enabled, set to ON, when you reset (*RST) or power up the instrument.

Syntax

```
SYSTem:PROTect
SYSTem:PROTect?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value 1

Errors and Events None

Dependencies None

Examples

Command: `SYST:PROT ON`

Query: `SYST:PROT?`

Response: 1

Related Commands

INPut:PROTection:STATe
SYSTem:SECurity:IMMediate
TEST:LOG:CLEar
*PUD

SYSTem:SECurity:IMMediate

Immediately destroys all measurement data and stored instrument settings. The instrument writes over measurement data in acquisition memory three times with a fixed bit pattern. Current settings are initialized to their *RST or factory default values but critical calibration constants are retained. Set SYSTem:PROTect to OFF before using the command SYSTem:SECurity:IMMediate.

Syntax

```
SYSTem:SECurity:IMMediate
```

Parameters None

Reset Value

Not applicable

Errors and Events

Execution Error –203, “Command protected”
Attempted to execute this command without first removing the protection with the command SYSTem:PROTect.

Dependencies

All measurement data and instrument settings are lost upon execution.

Examples

Command: `SYST:SEC:IMM`

Related Commands

*RST

SYSTem:SET
SYSTem:SET?

Sets or queries the internal state of the instrument as a binary data block. The format of the binary data is instrument specific and should not be modified if you intend to use it to set the instrument state. Binary settings may not be compatible with other models or firmware versions of the TVS600 Series Waveform Analyzers. Use the *LRN? query to get an ASCII version of the instrument settings.

Syntax

```
SYSTem:SET <settings>
SYSTem:SET?
```

Parameters

<settings>	Query response
<def_len_block> or <indef_len_block>	<def_len_block>

Reset Value

Not applicable

Errors and Events

Execution Error –233, “Invalid version”
Attempted to load a block of binary settings that have a version number different from that of the instrument.

Dependencies

None

Examples

Command: `SYST:SET #45000 . . .`

Query: `SYST:SET?`

Response: `#45000 . . .`

Related Commands

*LRN?
*RCL
*SAV

SYSTem:VERSion? (Query Only)

Returns the SCPI version supported by the waveform analyzer. You will need this version number if you call Tektronix for product or application support.

Syntax

```
SYSTem:VERSion?
```

Parameters

<version>	Query response
Not applicable	<NR2> 1995.0

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `SYST:VERS?`

Response: `1995.0`

Related Commands

*IDN?

TEST Subsystem

This section describes the commands in the TEST subsystem. See Figure 2–24. These commands execute the internal self-tests of the waveform analyzer module and return the pass or fail status.

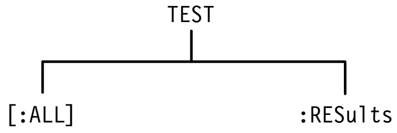


Figure 2–24: TEST subsystem hierarchy

TEST
TEST?

Executes all internal self-tests once. If a failure occurs, the test will stop. The query form returns the number of the first test that fails or zero if there were no failures. The command form executes the same tests but does not return a result value.

Use the command *WAI, *OPC, or *OPC? to delay execution of the TEST? command until the self test completes. Completion of the tests clears the TESTing flag in the Operation Status register.

Upon a failure, you should refer the instrument to your Service provider for repair. For a listing of the failure codes, refer to the *TVS600 Series Waveform Analyzers Service Manual*.

Syntax

```
TEST[:ALL]
TEST[:ALL]?
```

Parameters

None	Query response	
Not applicable	<NR1>	
	0	(no failure)
	1000 – 1999	(self test failure)

Reset Value

Not applicable

Errors and Events

None

Dependencies

Examples

Command: `TEST`

Query: `TEST?`

Response: `0`

Related Commands

TEST:RESults?
TEST:RESults:VERBose?

TEST:RESults? (Query Only)

Returns the failure code for the last self-test command that was executed. The returned failure code identifies the first test that failed. See the *Parameters* table for the possible responses. Use the command *WAI, *OPC, or *OPC? to delay execution of the TEST:RESults? command until the self test completes. Completion of the tests clears the TESTing flag in the Operation Status register.

Syntax

```
TEST:RESults[:CODE]?
```

Parameters

<failure_code>	Query response
Not applicable	<NR1> –1 (test in progress) 0 (no failure) 1000 – 1999 (Self Test failure)

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `TEST:RES?`

Response: 0

Related Commands

TEST
TEST:RESults:VERBose?

TEST:RESults:VERBose? (Query Only)

Returns a failure code as a string describing the last executed self-test command and the results of the self test. The returned failure code describes the first test that failed. See the *Parameters* table for the possible responses. Use the command *WAI, *OPC, or *OPC? to delay execution of the TEST:RESults? command until the self test completes. Completion of the tests clears the TESTing flag in the Operation Status register.

Syntax

```
TEST:RESults:VERBose?
```

Parameters

<test_results>	Query response
Not applicable	<string> error number: <NR1> –1 (test in progress) 0 (no failure) 1000 – 1999 (Self Test failure) verbose message: (error specific)

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `TEST:RES:VERB?`

Response: `1001,"TST_1 max=444 min=222 . . ."`

Related Commands

TEST
TEST:RESults?

TEST:STOP

TVS600A Models Only

This command aborts an active test sequence at the end of the currently executing test.

There is no associated query for this command.

If no test is running this command has no effect.

Syntax `TEST:STOP`

Parameters None

Suffix Units and Multipliers None

Reset Value Not applicable

Errors and Events None

Dependencies None

Examples Command: `TEST:STOP`

Related Commands None

TRACe Subsystem

This section describes the commands in the TRACe subsystem which store and retrieve acquisition and measurement results. See Figure 2–25 and Figure 2–26. The SENSe subsystem command DATA? provides another way to transfer acquisition records and CALC:DATA? provides an alternate way to transfer the results of calculations.

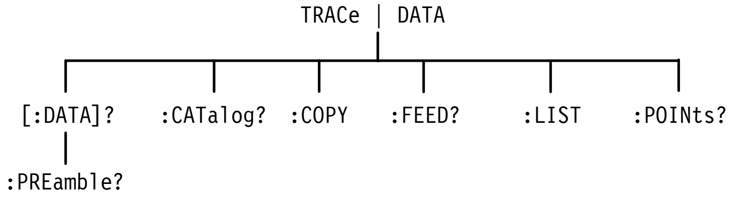


Figure 2–25: TRACe subsystem hierarchy

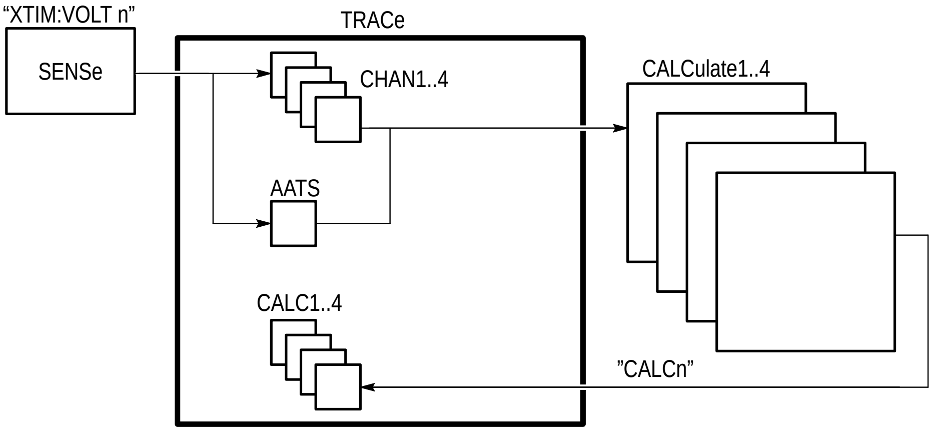


Figure 2–26: Functions of the TRACe subsystem

TRACe
TRACe?

The command form downloads waveforms or other data to the waveform analyzer. The query form uploads acquisition records or measurement results from the waveform analyzer to your VXIbus controller.

Command

TRACe sends the data and preamble (offset, scale, and other information) to the waveform analyzer. The data must be in a DIF (Data Interchange Format) to be accepted by the waveform analyzer.

Normally, you send both data and preamble, but you can send just the data or preamble block in your DIF block; if you do so, the rules described in Table 2–5 apply:

Table 2–5: Rules for Downloads

Download contains:	Destination REF exists	Destination REF doesn't exist
Preamble block and data	Overwrites existing preamble Overwrites existing data	Writes new preamble Writes new data
Preamble block, no data	Overwrites existing preamble Restructures existing data based on new preamble[1]	Writes new preamble Creates new data values initiated to zero
No preamble, data only	Existing preamble has size (only) adjusted to fit incoming data Overwrites existing data	Creates preamble with default scale (1.0), offset (0.0), and size (adjusted to fit incoming data) Writes new data

[1] **Size of existing reference trace will be truncated or zero-padded as necessary to match incoming preamble.**

Query

TRACe? returns the data in the format you specify. You set the data format for acquisition records and measurement results using the FORMat subsystem described on page 2–93. The query does not return data until pending acquisitions or calculations complete; therefore, the synchronizing commands *WAI, *OPC, and *OPC? are not required.

Single data records are returned unless auto-advance is on, in which case, the AADVance:RECord:STARt and :COUNt commands determine which and how many records are returned. This command returns raw data only; see the TRACe[:DATA]:PREamble? query to obtain scale and offset information.

Note the following points about data sources:

- Channels. The [SENSe:]DATA commands provide equivalent functions to the commands in the TRACe subsystem and there are fixed feeds between the SENSe and TRACe subsystems. [SENSe:]DATA? ”XTIM:VOLT 1” is equivalent to TRAC? CHAN1. The feed control for CHAN1..4 is set to ALWays (always enabled).

  [:DATA] is not accepted as a default node under DATA because the definition of [SENSe:]DATA? and DATA[:DATA]? would be ambiguous.

- CALC blocks. The CALC:DATA commands provide equivalent functions to the commands in the TRACe subsystem and there are fixed feed between the CALC and TRACE subsystems. CALC1:DATA? is equivalent to TRAC? CALC1. The feed control for CALC1..4 is set to ALWays (always enabled).

- AATS. The source AATS is the Auto Advance Time Stamp record. The record contains a sequence of times in seconds, from $t_{:STARt}$ to tn, between auto-advance records. The value $t_{:STARt}$ is the record specified with the AADV:REC:STAR command. Set the format of the AATS record to ASCII or 32 bit REAL numbers with the command FORM:TRAC:AATS. FDC transfers are binary only. The feed control for AATS is always enabled.

- References. REF1..10 are reference waveform records. There are no sense functions associated with these trace names, so there are no [SENSe:]DATA command equivalents.

For more information on uploading and downloading, refer to the waveform I/O discussion in the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
TRACe[:DATA] <destination>,(<block> | <dif_expression> |
             (<numeric_value>{,<numeric_value}))
TRACe[:DATA]? <source>
DATA:DATA <destination>,(<block> | <dif_expression> |
             (<numeric_value>{,<numeric_value}))
DATA:DATA? <source>
```

Parameters

<source>	Query response
CHAN1 CHAN2 CHAN3 CHAN4 AATS CALC1 CALC2 CALC3 CALC4 REF1 .. REF10	Not applicable

<destination>	Query response
REF1 .. REF10	Not applicable

<data>	Query response
Not applicable	Defined by FORMat:TRACe

Reset Value Not applicable

Errors and Events Execution Error –141, “Invalid character data”
Attempted to query data from an invalid trace name or attempted to send data to a destination other than REF1 though REF10.

Execution Error –200, “DIF error”
Attempted to send a DIF expression that is not recognized by the instrument.

Execution Error –223, “Too much data”
Attempted to send a data record larger than 64536 points.

Execution Error –230, “Data corrupt or stale”
Attempted to query data that is invalid, incomplete or stale.

Execution Error –241, “Hardware missing”
Attempted to query data from CHAN3 or CHAN4 when the instrument is configured with two channels.

Dependencies None

Examples Command: `TRAC REF1,<dif_block>`

Query: `TRAC? CHAN1`

Response: `<arb_block_data>`

Related Commands [SENSe:]DATA?
[SENSe:]DATA:PREamble?
CALCulate:DATA?
CALCulate:DATA:PREamble?
TRACe:PREamble?

TRACe:CATalog?

Returns list of the predefined trace names in your waveform analyzer. The trace names are quoted strings separated by commas. CHAN1 to 4 refer to SENSe functions “XTIM:VOLT n”. CALC1 to 4 refer to the results data from the four CALCulate blocks. AATS refers to the Auto Advance Time Stamp record. REF1 though REF10 refers to the reference waveform records.

The CHAN3 and CHAN4 trace names are not defined when the waveform analyzer has only two channels.

There is no associated command for this query.

Syntax

```
TRACe:CATalog?
DATA:CATalog?
```

Data Types

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `TRAC:CAT?`

Response:

```
"CHAN1","CHAN2","CHAN3","CHAN4","AATS","CALC1",
"CALC2","CALC3","CALC4"
"REF1","REF2","REF3","REF4",
"REF5","REF6","REF7","REF8",
"REF9","REF10"
```

Note that CHAN3 and CHAN4 are not defined for two channel instruments.

Related Commands

TRACe?
TRACe:PREamble?

TRACe:COPY

Copies acquisition or measurement data, or reference waveform records, to the outgoing Fast Data Channel (FDC), or to reference waveform records (REF1 to REF10). The CHAN1 to CHAN4 and CALC1 to CALC4 commands refer to the corresponding input channel or CALC block result. LIST includes all traces defined with the command TRACe:LIST. Traces with no data are copied as null blocks. No error is generated.

There is no associated query for this command.

For more information, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
TRACe:COPY <destination>, <source>
DATA:COPY <destination>, <source>
```

Parameters

<destination>	Query response
FDC1 REF1 .. REF10	Not applicable

<source>	Query response
CHAN1 CHAN2 CHAN3 CHAN4 AATS CALC1 CALC2 CALC3 CALC4 LIST[1] REF1 .. REF10	Not applicable

[1] **Fast Data Channel only**

Reset Value

Not applicable

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to copy data to or from an invalid trace name.

Execution Error –211, ”Settings conflict”
Attempted to transfer data from a trace LIST to a REF waveform.

Execution Error –241, “Hardware missing”
Attempted to copy data from CHAN3 or CHAN4 when the instrument is configured with two channels.

Dependencies

None

Examples

Command: `TRAC:COPY FDC1, CHAN1`

Related Commands

TRACE:LIST

TRACe:DELete

TVS600A Models Only

This command deletes the data contents of the named reference trace.

There is no associated query for this command.

Traces with no valid data are deleted. No error is generated.

Syntax

```
TRACe:DELete[:NAME] <destination>
DATA:DELete[:NAME] <destination>
```

Parameters

<destination>	Query response
REF1 .. REF10	Not applicable

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

Command Error –141, “Invalid character data”
Attempted to delete an invalid trace name.

Dependencies

None

Examples

Command: `TRAC:DEL REF1`

Related Commands

TRACe:COPY
TRACe:DATA

TRACe:DELete:ALL

TVS600A Models Only

This command deletes the data contents of all reference traces.

There is no associated query for this command.

Traces with no valid data are deleted. No error is generated.

Syntax

```
TRACe:DELete:ALL
DATA:DELete:ALL
```

Parameters

None

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `TRAC:DEL:ALL`

Related Commands

SYST:SEC:IMM
TRACe:COPY
TRACe:DATA

TRACe:FEED?

Returns the source of data for the pre-defined trace names.

There is no associated command for this query.

Syntax

```
TRACe:FEED? <source>
DATA:FEED? <source>
```

Parameters

<source>	Query response
CHAN1	"XTIM:VOLT 1"
CHAN2	"XTIM:VOLT 2"
CHAN3	"XTIM:VOLT 3"
CHAN4	"XTIM:VOLT 4"
AATS	"AADV"
CALC1	"CALC1"
CALC2	"CALC2"
CALC3	"CALC3"
CALC4	"CALC4"
REF1 .. REF10	"REF1" .. "REF10"

Suffix Units and Multipliers

None

Reset Value

Not applicable

Errors and Events

Execution Error –141, "Invalid character data"
Attempted to query the feed for an invalid trace name.

Dependencies

None

Examples

Query: `TRAC:FEED? CHAN1`

Response: `"XTIM:VOLT 1"`

Related Commands

TRACe?
TRACe:PREamble?

TRACe:LIST
TRACe:LIST?

Sets or queries the list of traces to transfer through the VXI Fast Data Channel at the completion of each INITiate command or when LIST is specified as the source for the TRACe:COPY command.

If a trace contains no trace data, a null block is copied for that trace and no error is generated.

Syntax

```
TRACe:LIST <list>
TRACe:LIST?
DATA:LIST <list>
DATA:LIST?
```

Parameters

<list>	Query response
CHAN1 CHAN2 CHAN3 CHAN4 AATS CALC1 CALC2 CALC3 CALC4 REF1 .. REF10 NONE	CHAN1 CHAN2 CHAN3 CHAN4 AATS CALC1 CALC2 CALC3 CALC4 REF1 .. REF10 NONE

Reset Value

NONE

Errors and Events

Command Error 108, “Parameter not allowed”
Attempted to assign more than 20 traces to the list.

Execution Error –141, “Invalid character data”
Attempted to include an invalid trace name in the list.

Dependencies

None

Examples

Command: `TRAC:LIST CHAN1,CALC1`

Query: `TRAC:LIST?`

Response: `CHAN1,CALC1`

Related Commands

TRACE:COPY

TRACe:POINts?

Returns the number of sample points in the acquisition record or CALCulate block record. The record length is set differently for each source:

- CHAN<n> is set by SWEep:POINts.
- AATS (Auto-advance Time Stamp) is set by AADVance:COUNt.
- CALC<n> depends on the record length of the source waveform(s) and the measurement or calculation selected.
- Auto Advance is set by AADVance:RECord:STARt and AADVance:RECord:COUNt.
- The query does not return until pending acquisitions/calculations are complete. No *WAI, *OPC, or *OPC? is required unless the user wishes to synchronize in a different manner.
- The record length of a CHANn trace depends on the value of SWEep:POINts.
- The record length of the AATS trace depends on the value of SWEep:AADVance:COUNt.
- The record length of a CALCn trace depends on the the record length of the source waveform(s) and the measurement being performed.
- The record length of a REFn trace depends on the record length of the stored data.
- If Auto Advance is on, the AADVance:RECord:STARt and :COUNt commands determine the number of points returned by this query.
- If a trace contains no valid data, a length of zero is returned. No error is generated.
- There is no associated command for this query.

If the record contains no data, a length of zero is returned.

Syntax

```
TRACe:POINts? <source>
DATA:POINts? <source>
```

Parameters

<source>	Query response
CHAN1 CHAN2 CHAN3 CHAN4 AATS CALC1 CALC2 CALC3 CALC4 REF1 .. REF10	<NR1>

Reset Value

Not applicable

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to query the record length of an invalid trace name.

Execution Error –241, “Hardware missing”
Attempted to query the record length of CHAN3 or CHAN4 when the instrument is configured with two channels.

Dependencies

None

Examples

Query: `TRAC:POIN? CHAN1`

Response: `1024`

Related Commands

TRACe?
TRACE:PREamble?
SWEep:POINts

TRACe:PREamble?

Transfers the data preamble for acquisition records or measurement results from the waveform analyzer to your VXIbus controller. The data preamble includes scaling and length information for the associated data record. The query does not return data until pending acquisitions or calculations complete. Hence, the synchronizing commands *WAI, *OPC, and *OPC? are not required.

The [SENSe:]DATA:PRE and CALC:DATA:PRE commands provide equivalent functions to the command TRACe:PRE. For example, [SENSe:]DATA:PRE? "XTIM:VOLT 1" is equivalent to TRAC? CHAN1 and CALC1:DATA:PRE? is equivalent to TRAC? CALC1. The feed controls for CHAN1 to CHAN4 and CALC1 to CALC4 are always enabled.

The source AATS is the Auto Advance Time Stamp record which is not supported by the SENSe subsystem. The feed control for AATS is always enabled.

For more information, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

There are fixed feeds between the SENSe and TRACe subsystems. [SENSe:]DATA:PRE? ”XTIM:VOLT 1” is equivalent to TRAC:PRE? CHAN1. The feed control for CHAN1..4 is set to ALWays.

There are fixed feeds between the CALCulate and TRACe subsystems. CALC1:DATA:PRE? is equivalent to TRAC:PRE? CALC1. The feed control for CALC1..4 is set to ALWays.

REF1..10 are the reference waveform records. There are no sense functions associated with these trace names.

[:DATA] is not accepted as a default node under DATA because the definition of [SENSe:]DATA:PRE? and DATA[:DATA]:PRE? would be ambiguous.

Syntax

```
TRACe[:DATA]:PREamble? <source>
DATA:DATA:PREamble? <source>
```

Parameters

<source>	Query response
CHAN1 CHAN2 CHAN3 CHAN4 AATS CALC1 CALC2 CALC3 CALC4 REF1 .. REF10	Not applicable

<preamble>	Query response
Not applicable	DIF Expression

Reset Value

Not applicable

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to query the preamble for an invalid trace name.

Execution Error –230, “Data corrupt or stale”
Attempted to query the preamble of data that is invalid, incomplete or stale.

Execution Error –241, “Hardware missing”
Attempted to query the preamble for CHAN3 or CHAN4 when the instrument is configured with two channels.

Dependencies

None

Examples

Query: `TRAC:PRE? CHAN1`

Response: (DIF Expression)

Related Commands

[SENSe:]DATA?
[SENSe:]DATA:PREamble?
CALCulate:DATA?
CALCulate:DATA:PREamble?
TRACe?

TRIGger[:A] Subsystem

This section describes the commands in the TRIGger[:A] subsystem. See Figure 2–27. These commands operate with the ARM, INITiate, TRIGger:B and ABORt subsystems to trigger acquisitions. The defined alias for the SCPI trigger :SEQuence[1] is A.

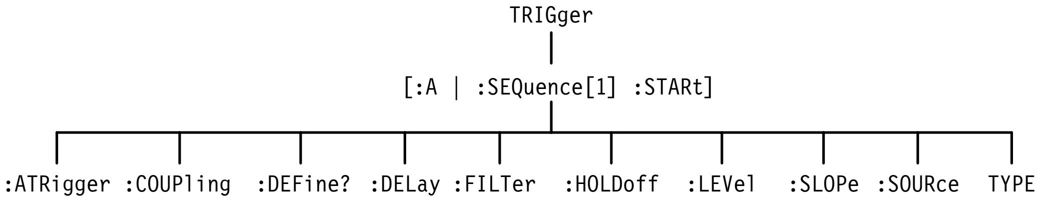


Figure 2–27: TRIGger:A (SCPI SEQuence1) subsystem hierarchy

TRIGger:ATRigger
TRIGger:ATRigger?

Sets or queries whether to generate an automatic trigger when the defined trigger does not occur within 500 ms. You should set TRIGger:ATRigger to OFF when acquiring signals of less than 2 Hz to avoid spurious automatic triggers. Automatic triggering works with all trigger sources.

Syntax

```
TRIGger[:A]:ATRigger[:STATe] <boolean>
TRIGger[:A]:ATRigger[:STATe]?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

None

Examples

Command: `TRIG:ATR ON`

Query: `TRIG:ATR?`

Response: 1

Related Commands

TRIGger:SOURce

TRIGger:COUPling
TRIGger:COUPling?

Sets or queries whether the source of the A trigger is AC or DC coupled. AC coupling removes any DC component from the signal. DC coupling passes all frequencies equally. This command is effective only when you set TRIGger:SOURce to an INTernal source. Note that the external trigger is limited to DC coupling only.

You may want to review the command TRIGger:COUPling:<preset> which lets you set trigger coupling and filtering with one command. However, that command does not adhere to the SCPI standard.

Syntax

```
TRIGger[:A]:COUPling <trigger_coupling>
TRIGger[:A]:COUPling?
```

Parameters

<trigger_coupling>	Query response
AC DC	AC DC

Reset Value

DC

Errors and Events

None

Dependencies

Changing trigger coupling from DC to AC sets TRIGger:FILTer[:LPASs] to OFF and may cause TRIGger:LEVel to change if the level is out of range for AC coupling. Changing trigger coupling from AC to DC sets TRIGger:FILTer:HPASs to OFF.

When TRIGger:A and TRIGger:B share the same SOURce, changing COUPling for one will change COUPling for the other.

Examples

Command: `TRIG:COUP AC`

Query: `TRIG:COUP?`

Response: `AC`

Related Commands

TRIGger:COUPling:<preset>
TRIGger:FILTer[:LPASs]
TRIGger:FILTer:HPASs

TRIGger:FILTer:NREJect
TRIGger:SOURce

TRIGger:COUPling:<preset>

Sets trigger coupling and filtering with one command. It should not be used in applications that must work on a variety of SCPI instruments. For a description of how this command affects the four underlying SCPI commands, see *Dependencies* on this page.

This command is effective only when you set TRIGger:SOURce to INTernal. Note that the external trigger provides DC coupling only. There is no associated query for this command.

Syntax

```
TRIGger[:A]:COUPling:AC
TRIGger[:A]:COUPling:ACNReject
TRIGger[:A]:COUPling:DC
TRIGger[:A]:COUPling:DCNReject
TRIGger[:A]:COUPling:HFReject
TRIGger[:A]:COUPling:LFReject
```

Parameters

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

The following table describes the interactions between TRIGger:COUPling:<preset> and its four underlying SCPI commands.

<preset>	COUPling	FILT[:LPASs]	FILT:HPAS	FILT:NREJect
AC	AC	OFF	OFF	OFF
ACNReject	AC	OFF	OFF	ON
DC	DC	OFF	OFF	OFF
DCNReject	DC	OFF	OFF	ON
HFReject	DC	ON	OFF	OFF
LFReject	AC	OFF	ON	OFF

Changing trigger coupling from DC to AC may cause TRIGger:LEVel to change if level is out of range for AC coupling.

When TRIGger:A and TRIGger:B share the same SOURce, changing COUPling for one will change COUPling for the other.

Examples Command: `TRIG:COUP:ACNR`

Related Commands TRIGger:COUPling
TRIGger:FILTer[:LPASs]
TRIGger:FILTer:HPASs
TRIGger:FILTer:NREJect

TRIGger:DEFine? (Query Only)

This query returns the predefined SEQuence1 alias, A. ARM:DEFine? and TRIGger:DEFine? return the same information.

Syntax

```
TRIGger[:SEQuence[1]]:DEFine?
```

Parameters

<sequence_alias>	Query response
Not applicable	<string> “A”

Reset Value “A”

Errors and Events None

Dependencies None

Examples

Query: `TRIG:DEF?`

Response: `"A"`

Related Commands ARM:DEFine?
TRIGger:SEQuence2:DEFine?

TRIGger:DELay TRIGger:DELay?

Sets or queries the trigger delay for the Trigger A circuit. Trigger delay sets the time after the trigger event to start acquisition. TRIGger:DELay is always positive which delays the start of acquisition a specified time after the trigger event. Use SWEep:OFFSet:TIME to acquire pretrigger data. The mininum real delay is 16 ns. Setting TRIGger:DELay to 0 indicates no delay and bypasses the delay counter.

For further information, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
TRIGger[:A]:DELay <delay_time>
TRIGger[:A]:DELay?
```

Parameters

<delay_time>[1]	Query response
<NRf> 16 ns ≤ N ≤ 250 s (in 4 ns steps)	<NR3>
MINimum	0.0E+0
MAXimum	250.0E+0

[1] **The default for the parameter <delay_time> is seconds (S). You can also use the multipliers MS for milliseconds, US for microseconds, and NS for nanoseconds.**

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set trigger delay to an illegal value.

Dependencies

Setting TRIGger:DELay to a value greater than 0 will cause TRIGger:B:DELay to be set to 0. Only one trigger delay may be active.

Setting TRIGger:DELay to a value greater than 0 when TRIGger:B:SOURce is set to INTernal<1234> or EXTernal, will set TRIGger:B:ECOunt to 1.

Examples

Command: `TRIG:DEL 10E-6`

Query: `TRIG:DEL?`

Response: `10.0E-6`

Related Commands

SWEep:OFFSet:TIME
TRIGger:B:DELay
TRIGger:B:ECOunt

TRIGger:FILTer[:LPASs]
TRIGger:FILTer[:LPASs]?

Sets or queries the state of the 50 kHz low pass trigger filter. Components of the trigger signal above 50 kHz are attenuated when the LPASs filter is on. It may only be used when DC coupled and with TRIGger:SOURce set to INTernal.

The INPut:FILTer is separate from the trigger filter and is located before the trigger pickoff in the signal path.

Syntax

```
TRIGger[:A]:FILTer[:LPASs][:STATe] <boolean>
TRIGger[:A]:FILTer[:LPASs][:STATe]?
```

Parameters

<boolean>	Query response
<NRf> N ≠ 0 or ON 0 or OFF	<NF1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

Enabling the low pass filter will set TRIGger:COUPling to DC and set TRIGger:FILTer:NREJect to OFF. When TRIGger:A and TRIGger:B share the same SOURce, changing FILTer for one will change FILTer for the other.

Examples

Command: `TRIG:FILT ON`

Query: `TRIG:FILT?`

Response: 1

Related Commands

TRIGger:COUPling
TRIGger:FILTer:HPASs
INPut:FILTer

TRIGger:FILTer:HPASs
TRIGger:FILTer:HPASs?

Sets or queries the state of the 50 kHz high pass trigger filter. The HPASs filter attenuates components of the trigger signal below 50 kHz. HPASs can only be used when AC coupled and with TRIGger:SOURce set to INTernal.

The INPut:FILTer is separate from the trigger filter and is located before the trigger pickoff in the signal path.

Syntax

```
TRIGger[:A]:FILTer:HPASs[:STATe] <boolean>
TRIGger[:A]:FILTer:HPASs[:STATe]?
```

Parameters

<boolean>	Query response
<NRf> N ≠ 0 or ON 0 or OFF	<NR1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

Enabling the high pass filter will set TRIGger:COUPling to AC and set TRIGger:FILTer:NREJect to OFF. When TRIGger:A and TRIGger:B share the same SOURce, changing FILTer for one will change FILTer for the other.

Examples

Command: `TRIG:FILT:HPAS ON`

Query: `TRIG:FILT:HPAS?`

Response: 1

Related Commands

TRIGger:COUPling
TRIGger:FILTer[:LPASs]
INPut:FILTer

TRIGger:FILTer:NREJect
TRIGger:FILTer:NREJect?

Sets or queries whether or not the noise reject filter is enabled. This filter provides a means of rejecting noise on the trigger signal. Only one trigger filter (i.e., LPAS, HPAS, or NREJ) may be enabled at a time. This command is effective only when you set TRIGger:SOURce to INTernal.

Syntax

```
TRIGger[:A]:FILTer:NREJect <boolean>
TRIGger[:A]:FILTer:NREJect?
```

Parameters

<boolean>	Query response
<NRf>	<NR1>
1 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

none

Dependencies

When TRIGger:A:SOURce and TRIGger:B:SOURce use the same signal, changing TRIG:FILT:NREJ also changes TRIG:B:FILT:NREJ.

Examples

Command: `TRIG:FILT:NREJ ON`

Query: `TRIG:FILT:NREJ?`

Response: 1

Related Commands

TRIGger:LEVel

TRIGger:HOLDoff:TIME
TRIGger:HOLDoff:TIME?

Sets or queries the trigger holdoff time. Trigger holdoff determines how long after one trigger event the event detector will ignore valid triggers. You can use holdoff to trigger at a particular point in a recurring sequence of pulses.

Syntax

```
TRIGger[:A]:HOLDoff:TIME <holdoff_time>
TRIGger[:A]:HOLDoff:TIME?
```

Parameters

<holdoff_time>[1]	Query response
<NRf>	<NR3>
250 ns ≤ N ≤ 12 s (in 8 ns steps)	
MINimum	250.0E–9
MAXimum	12.0E+0

[1] **The default for the parameter <holdoff_time> is seconds (S). You can also use the multipliers MS for milliseconds, US for microseconds, and NS for nanoseconds.**

Reset Value

250.0E–9

Errors and Events

Execution Error –222, "Data out of range"
Attempted to set holdoff time to an illegal value.

Dependencies

None

Examples

Command: `TRIG:HOLD:TIME 10E-6`

Query: `TRIG:HOLD:TIME?`

Response: `10.0E-6`

Related Commands

None

TRIGger:LEVel TRIGger:LEVel?

Sets or queries the trigger level for the TRIGger[:A] subsystem. The trigger level is a specific voltage through which the trigger signal must pass to be recognized as a trigger event and start acquisition. The trigger level should be within the range of the signal in order to guarantee a trigger event. TRIGger:LEVel is effective only when you set TRIGger:SOURce to INTernal or EXTernal. Note that the front panel TRIGD LED lights briefly when a trigger event occurs.

Attaching a probe modifies the minimum and maximum values as it does the setting for VOLTage:RANGe:PTPeak. Multiply the minimum and maximum limits by the attenuation factor of the probe to determine the new maximum and minimum values.

If you experience or expect an unstable trigger point due to noise on the trigger signal, use the noise reject feature of the trigger circuit to reduce the affects of noise. With the command TRIGger:FILTer:NREJect you can reduce the effects of noise on the internal trigger sources.

Syntax

```
TRIGger[:A]:LEVel <trigger_level>
TRIGger[:A]:LEVel?
```

Parameters

<trigger_level>[1]	Query response
<NRf>	<NR3>
External:	
–1.0 ≤ N ≤ 1.0 (2 mV steps)	
MINimum	–1.0E+0
MAXimum	1.0E+0
Internal DC Coupled:	
absolute maximum[2]: –200.0 ≤ N ≤ 200.0	VOLT:RANG:OFFS ± VOLT:RANG:PTP
limited by ± VOLT:RANG:PTP	
steps 0.002 * VOLT:RANG:PTP	
MINimum[2]	VOLT:RANG:OFFS – VOLT:RANG:PTP
MAXimum[2]	VOLT:RANG:OFFS + VOLT:RANG:PTP
Internal AC Coupled:	
absolute maximum[2]: –100.0 ≤ N ≤ 100.0	± VOLT:RANG:PTP
limited by ± VOLT:RANG:PTP	
steps 0.002 * VOLT:RANG:PTP	
MINimum[2]	– VOLT:RANG:PTP
MAXimum[2]	+ VOLT:RANG:PTP

[1] **The default for the parameter <trigger_level> is V for volts. You can also use the multipliers MV for millivolts and UV for microvolts.**

[2] **When you connect a probe, the maximum limits increase just as the vertical range increases for the input. For example, with a 1 V vertical range, connecting a 10X probe increases the vertical range and trigger level range to 10 V.**

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set trigger level to an illegal value.

Dependencies

None

Examples

Command: `TRIG:LEV 1`

Query: `TRIG:LEV?`

Response: `1.0E+0`

Related Commands

TRIGger:SOURce
VOLTage:RANGe:PTPeak
VOLTage:RANGe:OFFSet

TRIGger:METastable:STATe
TRIGger:METastable:STATe?

TVS600A Models Only

Sets or queries the state of metastable trigger rejection. Auto metastable rejection looks at the instrument context at acquisition time and determines whether to reject metastable triggers.

In Auto mode, metastable trigger rejection is enabled if averaging is on and AVERage:TYPe is SCALar or ENVelope but not PEAKdetect, or if any calc block contains AVER(), ENV(), or vector comparison functions.

Syntax

```
TRIGger:[A]:METastable:STATe <parameter>
TRIGger:[A]:METastable:STATe?
```

Parameters

<parameter>	Query response
REJect ACCept AUTo	

Suffix Units and Multipliers

Not applicable

Reset Value

AUTo

Errors and Events None

Dependencies None

Examples

Command: `TRIG:MET:STAT AUT`

Query: `TRIG:MET:STAT?`

Response: `AUT`

Related Commands None

TRIGger:SLOPe
TRIGger:SLOPe?

Sets or queries whether triggering occurs on the positive-going or negative-going edge of the trigger source. This command has effect only when TRIGger:SOURce is set to INTernal or EXTernal. The other trigger sources are digital signals.

Syntax

```
TRIGger[:A]:SLOPe <trigger_slope>
TRIGger[:A]:SLOPe?
```

Parameters

<trigger_slope>	Query response
POSitive	POS
NEGative	NEG

Reset Value POS

Errors and Events Execution Error –141, “Invalid character data”
Attempted to set slope to an illegal value.

Dependencies None

Examples

Command: `TRIG:SLOP NEGATIVE`

Query: `TRIG:SLOP?`

Response: `NEG`

Related Commands TRIGger:LEVel

TRIGger:SOURce
TRIGger:SOURce?

Sets or queries the source of the trigger signal for the trigger A circuit. You can specify only one source at a time.

Setting source to INTernal1 selects the signal from INPut1 and [SENSe:]VOLTage1 as the trigger source. The ECLTrg and TTLTrg sources are digital trigger signals from the VXI P2 backplane. The EXTernal signal is from the front-panel BNC connector labeled External Trigger Input.

Many commands in the TRIGger subsystem are dependent on the trigger source you select. For instance, the TRIGger:LEVel setting is ignored when you set the trigger source to TTLTrg0. The dependencies for each command are listed with the command description.

Syntax

```
TRIGger[:A]:SOURce <trigger_source>
TRIGger[:A]:SOURce?
```

Parameters

<trigger_source>	Query response
ECLTrg0	ECLT0
ECLTrg1	ECLT1
EXTernal	EXT
INTernal1	INT1
INTernal2	INT2
INTernal3[1]	INT3
INTernal4[1]	INT4
TTLTrg0	TTLT0
. . .	. . .
TTLTrg7	TTLT7

[1] **The selections INTernal3 and INTernal4 are not valid with the TVS621 and TVS625.**

Reset Value INT1

Errors and Events

Execution Error –141, "Invalid character data"
Attempted to set trigger source to IMMediate or another illegal value.

Execution Error –241, "Hardware missing"
Attempted to set trigger source to INTernal3 or INTernal4
when the instrument has two channels.

Dependencies

Changing the trigger source from INTernal to EXTernal or from one INTernal source to another INTernal source with a different VOLTage:RANGe setting may change the TRIGger:LEVel setting to accommodate a new vertical range.

Examples

Command: `TRIG:SOUR INT1`

Query: `TRIG:SOUR?`

Response: `INT1`

Related Commands

ARM:SOURce
TRIGger:B:SOURce

TRIGger:TYPE
TRIGger:TYPE?

Sets or queries the type of triggering to use for the next acquisition. The available types are as follows:

- EDGE specifies the default triggering. A trigger event occurs when a signal passes through a specified voltage level on the specified rising or falling slope. The standard SCPI trigger commands control this mode.
- LOGic–A trigger event occurs when the specified combination of input signals occur. (TVS600A models only)
- PULse generates a trigger event when a specified pulse is detected. It is controlled by commands in the TRIGger[:A]:PULSe subsystem.
- SHOLdtime–A trigger event occurs when the specified clock-to-data setup and hold time is violated. (TVS600A models only)
- TRANsition–A trigger event occurs when the input signal fails to transition between two thresholds in the specified time. (TVS600A models only)

Syntax

```
TRIGger[:A]:TYPE <trigger_type>
TRIGger[:A]:TYPE?
```

Parameters

<trigger_type>	Query response
EDGE	EDGE
LOGic	LOG
PULSe	PULS
SHOLdtime	SHOL
TRANsition	TRAN

Reset Value

EDGE

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set trigger type to an illegal value.

Dependencies

Setting TRIGger:TYPE to a value other than EDGE will set TRIGger:B:SOURce to IMMediate.

Examples

Command: `TRIG:TYPE PULS`

Query: `TRIG:TYPE?`

Response: `PULS`

Related Commands

None

TRIGger:B Subsystem

This section describes the commands in the TRIGger:B subsystem. See Figure 2–28. These commands operate with the ARM, INITiate, TRIGger[:A] and ABORt subsystems to trigger acquisitions. The TRIGger:B subsystem is an alias for the SCPI SEQuence2 trigger block. The defined alias for the SCPI trigger :SEQuence[2] is :B.

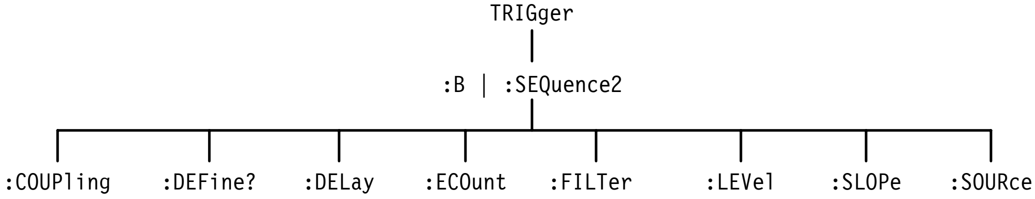


Figure 2–28: TRIGger:B (SCPI SEQuence2) subsystem hierarchy

TRIGger:B:COUPling
TRIGger:B:COUPling?

Sets or queries whether the source of the B trigger is AC or DC coupled. AC coupling removes any DC component from the signal. DC coupling passes all frequencies equally. This command is effective only when you set TRIGger:SOURce to an INTernal source. Note that the external trigger is limited to DC coupling only.

You may want to review the command TRIGger:COUPling:<preset> which lets you set trigger coupling and filtering with one command. However, that command does not adhere to the SCPI standard.

Syntax

```
TRIGger:B:COUPling <trigger_coupling>
TRIGger:B:COUPling?
```

Parameters

<trigger_coupling>	Query response
AC DC	AC DC

Reset Value

DC

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set coupling to an illegal value.

Dependencies

Changing B trigger coupling from DC to AC sets TRIGger:B:FILTer[:LPASs] to OFF and may cause TRIGger:B:LEVel to change if the level is out range for AC coupling.

Changing B trigger coupling from AC to DC sets TRIGger:B:FILTer:HPASs to OFF.

Examples

Command: `TRIG:B:COUP AC`

Query: `TRIG:B:COUP?`

Response: `AC`

Related Commands

TRIGger:B:COUPling:<preset>
TRIGger:B:FILTer[:LPASs]
TRIGger:B:FILTer:HPASs

TRIGger:B:FILTer:NREJect
TRIGger:B:SOURce

TRIGger:B:COUPling:<preset>

This command provides a traditional way of setting the B trigger coupling and filtering in one step. It should not be used in applications that must work on a variety of SCPI instruments. For a description of how this command affects the four underlying SCPI commands, see *Dependencies* for this command.

This command is effective only when you set TRIGger:B:SOURce to INTernal. Note that the external trigger provides DC coupling only. There is no associated query for this command.

Syntax

```
TRIGger:B:COUPling:AC
TRIGger:B:COUPling:ACNReject
TRIGger:B:COUPling:DC
TRIGger:B:COUPling:DCNReject
TRIGger:B:COUPling:HFReject
TRIGger:B:COUPling:LFReject
```

Parameters

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

The following table describes the interactions between TRIGger:B:COUPling:<preset> and its four underlying SCPI commands.

<preset>	COUPling	FILT[:LPASs]	FILT:HPASs	FILT:NREJect
AC	AC	OFF	OFF	OFF
ACNReject	AC	OFF	OFF	ON
DC	DC	OFF	OFF	OFF
DCNReject	DC	OFF	OFF	ON
HFReject	DC	ON	OFF	OFF
LFReject	AC	OFF	ON	OFF

Changing trigger coupling from DC to AC may cause TRIGger:B:LEVel to change if level is out of range for AC coupling.

Examples

Command: `TRIG:B:COUP:ACNR`

Related Commands

TRIGger:COUPling
TRIGger:FILTer[:LPASs]
TRIGger:FILTer:HPASs
TRIGger:FILTer:NREJect

TRIGger:SEQuence2:DEFine? (Query Only)

This query returns the predefined SEQuence2 alias, B.

Syntax

```
TRIGger:SEQuence2:DEFine?
```

Parameters

<sequence_alias>	Query response
Not applicable	<string> “B”

Reset Value

“B”

Errors and Events

None

Dependencies

None

Examples

Query: `TRIG:SEQ2:DEF?`

Response: `"B"`

Related Commands

ARM:DEFine?
TRIGger:DEFine?

TRIGger:B:DELay
TRIGger:B:DELay?

Sets or queries the trigger delay for the Trigger B circuit. Trigger delay sets the time after the Trigger B event to start acquisition. TRIGger:B:DELay is always positive which delays the start of acquisition a specified time after the trigger event. The minimum real delay is 16 ns. Setting TRIGger:DELay to 0 indicates no delay and bypasses the delay counter. Use SWEep:OFFSet:TIME to acquire pretrigger data.

For further information, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
TRIGger:B:DELay <delay_time>
TRIGger:B:DELay?
```

Parameters

<delay_time>[1]	Query response
<NRf> 16 ns ≤ N ≤ 250 s (in 4 ns steps)	<NR3>
MINimum MAXimum	0.0E+0 250.0E+0

[1] **The default for the parameter <delay_time> is seconds (S). You can also use the multipliers MS for milliseconds, US for microseconds, and NS for nanoseconds.**

Reset Value

0.0E+0

Errors and Events

Execution Error –222, "Data out of range"
Attempted to set B trigger delay to an illegal value.

Dependencies

Setting TRIGger:B:DELay to a value greater than 0 will cause TRIGger[:A]:DELay to be set to 0. Only one of the delays may operate at a time.

Examples

Command: `TRIG:B:DEL 10E-6`

Query: `TRIG:B:DEL?`

Response: `10.0E-6`

Related Commands

SWEep:OFFSet:TIME
TRIGger[:A]:DELay
TRIGger:B:ECOunt

TRIGger:B:ECOunt
TRIGger:B:ECOunt?

Sets or queries the number of B trigger events to count before starting acquisition. The number of B trigger events is ignored when you set TRIGger:B:SOURce to IMMediate.

Syntax

```
TRIGger:B:ECOunt <delay_events>
TRIGger:B:ECOunt?
```

Parameters

<delay_events>	Query response
<NRf> 1 ≤ N ≤ 10,000,000	<NR1>
MINimum	1
MAXimum	10000000

Reset Value

1

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set B trigger events to an illegal value.

Dependencies

Setting TRIGger:B:ECOunt to a value greater than 1 when TRIGger:B:SOURce is set to INTernal or EXTernal sets TRIGger[:A]:DELay to 0.

Examples

Command: `TRIG:B:ECO 100`

Query: `TRIG:B:ECO?`

Response: `100`

Related Commands

TRIGger[:A]:DELay
TRIGger:B:DELay

TRIGger:B:FILTer[:LPASs]
TRIGger:B:FILTer[:LPASs]?

Sets or queries the state of the 50 kHz, low-pass filter for the B trigger circuit. Components of the trigger signal above 50 kHz are attenuated when the LPASs filter is on. It may only be used when DC coupled and with TRIGger:SOURce set to INTernal. Only one trigger filter (i.e., LPAS, HPAS, or NREJ) may be enabled at a time.

The INPut:FILTer is separate from the trigger filter and is located before the trigger pickoff in the signal path.

Syntax

```
TRIGger:B:FILTer[:LPASs][:STATe] <boolean>
TRIGger:B:FILTer[:LPASs][:STATe]?
```

Parameters

<boolean>	Query response
<NRf>	<NR1>
1 or ON	1
0 or OFF	0

Reset Value

0

Errors and Events

None

Dependencies

Enabling the low pass filter will set TRIGger:B:COUPling to DC and set TRIGger:B:FILTer:NREJect to OFF.

When TRIGger:A:SOURce and TRIGger:B:SOURce are set to the same signal, changing LPAS for one will change LPAS for the other.

Examples

Command: `TRIG:B:FILT ON`

Query: `TRIG:B:FILT?`

Response: 1

Related Commands

TRIGger:B:COUPling
TRIGger:B:FILTer:HPASs
INPut:FILTer

TRIGger:B:FILTer:HPASs
TRIGger:B:FILTer:HPASs?

Sets or queries the state of the 50 kHz high pass filter for the B trigger circuit. The HPASs filter attenuates components of the trigger signal below 50 kHz. HPASs can only be used when AC coupled and with TRIGger:SOURce set to INTernal. Only one trigger filter (i.e., LPAS, HPAS, or NREJ) may be enabled at a time.

The INPut:FILTer is separate from the trigger filter and is located before the trigger pickoff in the signal path.

Syntax

```
TRIGger:B:FILTer:HPASs[:STATe] <boolean>
TRIGger:B:FILTer:HPASs[:STATe]?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

Enabling the high pass filter sets TRIGger:B:COUPling to AC and TRIGger:B:FILTer:NREJect to OFF.

When TRIGger:A:SOURce and TRIGger:B:SOURce are set to the same signal, changing HPAS for one will change HPAS for the other.

Examples

Command: `TRIG:B:FILT:HPAS ON`

Query: `TRIG:B:FILT:HPAS?`

Response: 1

Related Commands

TRIGger:B:COUPling
TRIGger:B:FILTer[:LPASs]
TRIGger:B:FILTer:NREJect
INPut:FILTer

TRIGger:B:FILTer:NREJect
TRIGger:B:FILTer:NREJect?

Sets or queries whether or not the noise reject filter is enabled. This filter provides a means of rejecting noise on the trigger signal. Only one trigger filter (i.e., LPAS, HPAS, or NREJ) may be enabled at a time. This command is effective only when you set TRIGger:SOURce to INTernal

Syntax

```
TRIGger:B:FILTer:NREJect <boolean>
TRIGger:B:FILTer:NREJect?
```

Parameters

<boolean>	Query response
<NRf> 1 or ON 0 or OFF	<NR1> 1 0

Reset Value

0

Errors and Events

None

Dependencies

Enabling NREJ sets HPAS and LPAS trigger filters to OFF.

When TRIGger:A:SOURce and TRIGger:B:SOURce are set to the same signal, changing NREJ for one will change NREJ for the other.

Examples

Command: `TRIG:B:FILT:NREJ ON`

Query: `TRIG:B:FILT:NREJ?`

Response: 1

Related Commands

TRIGger:B:LEVel

TRIGger:B:LEVel
TRIGger:B:LEVel?

Sets or queries the trigger level for the TRIGger:B subsystem. The trigger level is a specific voltage through which the trigger signal must pass to be recognized as a trigger event and start acquisition. The trigger level should be within the range of the signal in order to guarantee a trigger event. TRIGger:LEVel is effective only when you set TRIGger:SOURce to INTernal or EXTernal. Note that the front panel TRIGD LED lights briefly when a trigger event occurs.

Attaching a probe modifies the minimum and maximum values as it does the setting for VOLTage:RANGe:PTPeak. Multiply the minimum and maximum limits by the attenuation factor of the probe to determine the new maximum and minimum values.

If you experience or expect an unstable trigger point due to noise on the trigger signal, use the noise reject feature of the trigger circuit to reduce the affects of noise. With the command TRIGger:B:FILTer:NREJect you can reduce the effects of noise on the internal trigger sources.

Syntax

```
TRIGger:B:LEVel <trigger_level>
TRIGger:B:LEVel?
```

Parameters

<trigger_level>[1]	Query response
<NRf> External: –1.0 ≤ N ≤ 1.0 (2 mV steps) MINimum MAXimum	<NR3> –1.0E+0 1.0E+0
Internal DC Coupled: absolute maximum[2]: –200.0 ≤ N ≤ 200.0 limited by ± VOLT:RANG:PTP steps 0.002 * VOLT:RANG:PTP MINimum MAXimum	 VOLT:RANG:OFFS ± VOLT:RANG:PTP VOLT:RANG:OFFS – VOLT:RANG:PTP VOLT:RANG:OFFS + VOLT:RANG:PTP
Internal AC Coupled: absolute maximum: –100.0 ≤ N ≤ 100.0 limited by ± VOLT:RANG:PTP steps 0.002 * VOLT:RANG:PTP MINimum MAXimum	 ± VOLT:RANG:PTP – VOLT:RANG:PTP + VOLT:RANG:PTP

[1] **The default for the parameter <trigger_level> is V for volts. You can also use the multipliers MV for millivolts and UV for microvolts.**

[2] **When you connect a probe, the maximum limits increase just as the vertical range increases for the input. For example, with a 1 V vertical range, connecting a 10X probe increases the vertical range and trigger level range to 10 V.**

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set B trigger level to an illegal value.

Dependencies

None

Examples

Command:	`TRIG:B:LEV 1`
Query:	`TRIG:B:LEV?`
Response:	`1.0E+0`

Related Commands

TRIGger:B:FILTer:NREJect
TRIGger:B:SOURce
VOLTage:RANGe:PTPeak
VOLTage:RANGe:OFFSet

TRIGger:B:SLOPe
TRIGger:B:SLOPe?

Sets or queries whether triggering occurs on the positive-going or negative-going edge of the trigger B signal source. This command has effect only when TRIGger:B:SOURce is set to INTernal or EXTernal. The other trigger sources are digital signals.

Syntax

```
TRIGger:B:SLOPe <trigger_slope>
TRIGger:B:SLOPe?
```

Parameters

<trigger_slope>	Query response
POSitive	POS
NEGative	NEG

Reset Value

POS

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set slope to an illegal value.

Dependencies

None

Examples

Command: `TRIG:B:SLOP NEGATIVE`

Query: `TRIG:B:SLOP?`

Response: `NEG`

Related Commands

TRIGger:B:LEVel

TRIGger:B:SOURce
TRIGger:B:SOURce?

Sets or queries the source of the trigger signal for the trigger B circuit. You can specify only one source at a time.

Setting source to INTernal1 selects the signal from INPut1 and [SENSe:]VOLTage1 as the trigger source. The EXTernal signal is from the front-panel BNC connector labeled External Trigger Input. Setting the source to IMMediate will bypass event detection in the trigger B circuit forcing a trigger B event.

Many commands in the TRIGger subsystem are dependent on the trigger source you select. For instance, the TRIGger:B:COUPling setting is ignored when you set the trigger source to EXTernal. The dependencies for each command are listed with the command description.

Syntax

```
TRIGger:B:SOURce <trigger_source>
TRIGger:B:SOURce?
```

Parameters

<trigger_source>	Query response
EXTernal	EXT
IMMediate	IMM
INTernal1	INT1
INTernal2	INT2
INTernal3	INT3
INTernal4	INT4

Reset Value

IMM

Errors and Events

Execution Error –141, “Invalid character data”
Attempted to set the trigger source to an illegal value.

Execution Error –221, “Settings conflict.”
Attempted to set trigger source to a value other than IMMediate when TRIGger:TYPE is set to PULSe.

Execution Error –241, “Hardware missing”
Attempted to set the trigger source to INTernal3 or INTernal4 when the instrument has two channels.

Dependencies

Changing the trigger B source from INTernal to EXTernal or from one INTernal source to another INTernal source with a different VOLTage:RANGe setting may change the TRIGger:B:LEVel setting to accommodate a new vertical range.

Setting TRIGger:B:SOURce to INTernal[1, 2, 3, 4] or EXTernal when TRIGger:B:ECOunt is greater than 1 sets TRIGger[:A]:DELay to 0.

Examples

Command: `TRIG:B:SOUR INTERNAL1`

Query: `TRIG:B:SOUR?`

Response: `INT1`

Related Commands

ARM:SOURce
TRIGger[:A]:SOURce

TRIGger:LOGic Subsystem (TVS600A Models Only)

This section describes each command and query in the TRIGger:LOGic subsystem. See Figure 2–29. These commands are used to control pattern and state triggering.

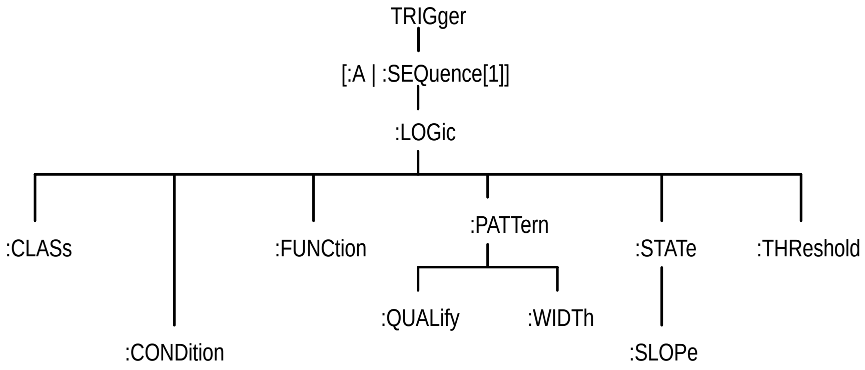


Figure 2–29: TRIGger:LOGic subsystem hierarchy

TRIGger:LOGic:CLASs
TRIGger:LOGic:CLASs?

TVS600A Models Only

Sets or queries the class of logic triggering selected. PATTern: the instrument triggers when the specified logical combination of channels 1, 2, 3 and 4 are met. STATe: the instrument triggers when the specified conditions of channels 1, 2 and 3 are met at the same time a transition occurs on channel 4.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is LOGic.

TRIGger:LOGic:CLASs STATE is not available when the instrument is configured with two channels.

Syntax

```
TRIGger[:A]:LOGic:CLASs <logic_class>
TRIGger[:A]:LOGic:CLASs?
```

Parameters

<logic_class>	Query response
PATTern	PATT
STATe	STAT

Suffix Units and Multipliers

None

Reset Value

PATT

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set class to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to set class to STATe when the instrument is configured with two channels.

Dependencies

None

Examples

Command: `TRIG:LOG:CLAS STATe`

Query: `TRIG:LOG:CLAS?`

Response: `STAT`

Related Commands TRIGger:TYPE

TRIGger:LOGic:CONDition
TRIGger:LOGic:CONDition?

TVS600A Models Only

Sets or queries the trigger condition for each channel when logic triggering is selected. The pattern argument refers to the two or four channels of the instrument in a left to right syntax for channel 4(2) through channel 1. A “1” indicates a High, a “0” indicates a Low, and an “X” indicates a Don’t Care is selected.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is set to LOGic.

The condition of channel 4 is ignored when TRIGger:LOGic:CLASs is set to STATe.

The condition of Bits 3 and 4 are ignored when the instrument is configured with two channels.

Syntax

```
TRIGger[:A]:LOGic:CONDition <pattern>
TRIGger[:A]:LOGic:CONDition?
```

Parameters

<paggern>	Query response
LCXXX0	LCXXX0
LCXXX1	LCXXX1
....	
LCM1111	LCM1111

Suffix Units and Multipliers None

Reset Value LC0000

Errors and Events Command Error –141, “Invalid character data”
Attempted to set either condition parameter to an illegal value.

Dependencies None

Examples Command: `TRIG:LOG:COND LC1100`

Query: `TRIG:LOG:COND?`

Response: `LC1X10`

Related Commands

TRIGger:TYPE
TRIGger:LOGic:CLASs

TRIGger:LOGic:FUNCtion
TRIGger:LOGic:FUNCtion?

TVS600A Models Only

Sets or queries the logical combination of the input channels for the main logic trigger. When TRIGger:LOGic:CLASs is set to PATTern, this command applies to channel 1, 2, 3 and 4. When TRIGger:LOGic:CLASs is set to STATe, only channels 1, 2 and 3 are logically combined.

:A is a pre–defined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is set to LOGic.

Syntax

```
TRIGger[:A]:LOGic:FUNCtion <logic_func>
TRIGger[:A]:LOGic:FUNCtion?
```

Parameters

<logic_fun>	Query response
AND	AND
NAND	NAND
NOR	NOR
OR	OR

Suffix Units and Multipliers

None

Reset Value

AND

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set FUNCtion to an illegal value.

Dependencies

None

Examples

Command: `TRIG:LOG:FUNC NOR`

Query: `TRIG:LOG:FUNC?`

Response: `NOR`

Related Commands

TRIGger:TYPE
TRIGger:LOGic:CLASs

TRIGger:LOGic:PATTern:QUALify TRIGger:LOGic:PATTern:QUALify?

TVS600A Models Only

Sets or queries the type of time qualification performed when pattern logic triggering is selected. OFF means that there is no time qualification. LT means that the trigger occurs when the pattern is true less than the specified width. GT means that the trigger occurs when the pattern is true greater than the specified width.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is set to LOGic and TRIGger:LOGic:CLASs is PATTern.

Syntax

```
TRIGger[:A]:LOGic:PATTern:QUALify <type>
TRIGger[:A]:LOGic:PATTern:QUALify?
```

Parameters

<type>	Query response
GT LT OFF	GT LT OFF

Suffix Units and Multipliers

None

Reset Value

OFF

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set QUALify to an illegal value.

Dependencies

None

Examples

Command: `TRIG:LOG:PATT:QUAL GT`

Query: `TRIG:LOG:PATT:QUAL?`

Response: `GT`

Related Commands

TRIGger:TYPE
TRIGger:LOGic:CLASs
TRIGger:LOGic:PATTern:WIDTh

TRIGger:LOGic:PATTern:WIDTh
TRIGger:LOGic:PATTern:WIDTh?

TVS600A Models Only

Sets or queries the width of the pattern when time qualification is enabled.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is LOGic, TRIGger:LOGic:CLASs is PATTern, and TRIGger:LOGic:PATTern:QUALify is LT or GT.

Syntax

```
TRIGger[:A]:LOGic:PATTern:WIDTh <time>
TRIGger[:A]:LOGic:PATTern:WIDTh?
```

Parameters

<time>	Query response
<NRf>	<NR3>
MINimum MAXimum	2.0E–9 1.0E+0

Suffix Units and Multipliers

S (default), MS, US, NS, or PS

Reset Value

5.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set WIDTh to an illegal value.

Dependencies

None

Examples

Command: `TRIG:LOG:PATT:WIDT 0.001`

Query: `TRIG:LOG:PATT:WIDT?`

Response: `1.0E-3`

Related Commands

TRIGger:TYPE
TRIGger:LOGic:CLASs
TRIGger:LOGic:PATTern:QUALify

TRIGger:LOGic:STATe:SLOPe
TRIGger:LOGic:STATe:SLOPe?

TVS600A Models Only

Sets or queries the slope of the clock (Channel 4) for logic state triggering.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is LOGic and TRIGger:LOGic:CLASs is STATe.

Syntax

```
TRIGger[:A]:LOGic:STATe:SLOPe <slope>
TRIGger[:A]:LOGic:STATe:SLOPe?
```

Parameters

<slope>	Query response
NEGative	NEG
POSitive	POS

Suffix Units and Multipliers

None

Reset Value

POS

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set SLOPe to an illegal value.

Dependencies

None

Examples

Command: `TRIG:LOG:STAT:SLOP NEG`

Query: `TRIG:LOG:STAT:SLOP?`

Response: `NEG`

Related Commands

TRIGger:TYPE
TRIGger:LOGic:CLASs

TRIGger:LOGic:THReshold
TRIGger:LOGic:THReshold?

TVS600A Models Only

Sets or queries the trigger threshold for each channel when logic triggering is selected.

:A is a predefined alias for :SEQuence[1].

The minimum and maximum values for threshold are VOLT:RANG:OFFS +/– VOLT:RANG:PTP (in VOLT:RANG:PTP * 0.002 steps).

This command only has effect when TRIGger:TYPE is LOGic.

TRIGger:LOGic:THReshold1 is associated with INPut1, etc.

Syntax

```
TRIGger[:A]:LOGic:THReshold[1] <level>
TRIGger[:A]:LOGic:THReshold[1]?
TRIGger[:A]:LOGic:THReshold2 <level>
TRIGger[:A]:LOGic:THReshold2?
TRIGger[:A]:LOGic:THReshold3 <level>
TRIGger[:A]:LOGic:THReshold3?
TRIGger[:A]:LOGic:THReshold4 <level>
TRIGger[:A]:LOGic:THReshold4?
```

Parameters

<level>	Query response
<NRf>	<NR3>
MINimum MAXimum	<See comments> <See comments>

Suffix Units and Multipliers

V (default), MV, or UV

Reset Value

0.0V

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set threshold to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to program THReshold3 or THReshold4
when the instrument is configured with two channels.

Dependencies

None

Examples

Command: `TRIG:LOG:THR2 1.0`

Query: `TRIG:LOG:THR2?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE

TRIGger:PULSe Subsystem

This section describes the commands in the TRIGger:PULSe subsystem. See Figure 2–30. The pulse trigger commands provide the capability to trigger when a pulse occurs that is outside specified parameters.

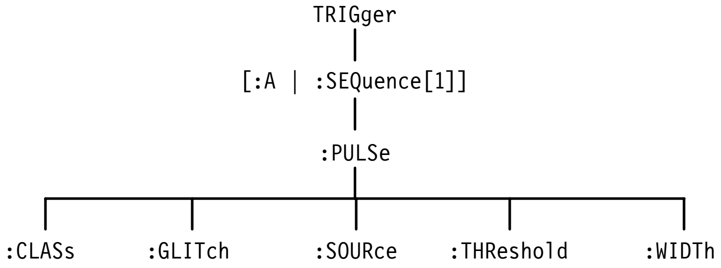


Figure 2–30: TRIGger:PULSe subsystem hierarchy

TRIGger:PULSe:CLASs
TRIGger:PULSe:CLASs?

Sets or queries the class of pulse triggering to use for the next acquisition. The trigger classes are as follows:

- GLITch — the trigger event occurs when the instrument detects a pulse that is either narrower or wider than a specified duration or width.
- TIMEout —the trigger event occurs when the pulse width timer expires on pulses wider than the specified duration
- WIDTh — the trigger event occurs when the instrument detects a pulse that is either inside or outside a specified duration or width.

This command has effect only when you set TRIGger:TYPE to PULSe. The defined alias for the SCPI trigger :SEQuence[1] is A.

Syntax

```
TRIGger[:A]:PULSe:CLASs <pulse_class>
TRIGger[:A]:PULSe:CLASs?
```

Parameters

<pulse_class>	Query response
GLITch	GLIT
TIMEout[1]	TIME
WIDTh	WIDT

[1] **TVS600A models only**

Reset Value

GLITch

Errors and Events

Execution Error –224, “Illegal parameter value”
Attempted to set class to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:CLAS WIDTH`

Query: `TRIG:PULS:CLAS?`

Response: `WIDT`

Related Commands

TRIGger:TYPE

TRIGger:PULSe:GLITch:POLarity TRIGger:PULSe:GLITch:POLarity?

Sets or queries the polarity of the event pulse for pulse glitch triggering. Before this command will be effective, you must set TRIGger:TYPE to PULSe and TRIGger:PULSe:CLASs to GLITch. Refer to the discussion of TRIGger:PULSe:CLASs on page 2–270.

Syntax

```
TRIGger[:A]:PULSe:GLITch:POLarity <polarity>
TRIGger[:A]:PULSe:GLITch:POLarity?
```

Parameters

<polarity>	Query response
EITHer	EITH
NEGative	NEG
POSitive	POS

Reset Value

POS

Errors and Events

Execution Error –224, "Illegal parameter value"
Attempted to set polarity to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:GLIT:POL NEG`

Query: `TRIG:PULS:GLIT:POL?`

Response: `NEG`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs

TRIGger:PULSe:GLITch:QUALify
TRIGger:PULSe:GLITch:QUALify?

Sets or queries the type of time qualification for pulse glitch triggering. LT specifies that the trigger event occurs when the glitch width is less than the specified time. GT specifies that the trigger event occurs when the glitch width is greater than the specified time.

Before this command will be effective, you must set TRIGger:TYPE to PULSe and TRIGger:PULSe:CLASs to GLITch. Refer to the discussion of TRIGger:PULSe:CLASs on page 2–270.

Syntax

```
TRIGger[:A]:PULSe:GLITch:QUALify <type>
TRIGger[:A]:PULSe:GLITch:QUALify?
```

Parameters

<type>	Query response
GT	GT
LT	LT

Reset Value

LT

Errors and Events

Execution Error –224, “Illegal parameter value”
Attempted to set qualify to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:GLIT:QUAL GT`

Query: `TRIG:PULS:GLIT:QUAL?`

Response: `GT`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs
TRIGger:PULSe:GLITch:WIDTh

TRIGger:PULSe:GLITch:WIDTh
TRIGger:PULSe:GLITch:WIDTh?

Sets or queries the width of the pulse used for pulse glitch triggering. Use the command TRIGger:PULSe:GLITch:QUALify to set whether the trigger event is for pulses less than or greater than the WIDTh you specify.

Before this command will be effective, you must set TRIGger:TYPE to PULSe and TRIGger:PULSe:CLASs to GLITch. Refer to the discussion of TRIGger:PULSe:CLASs on page 2–270. The defined alias for the SCPI trigger :SEQuence[1] is A.

Syntax

```
TRIGger[:A]:PULSe:GLITch:WIDTh <width>
TRIGger[:A]:PULSe:GLITch:WIDTh?
```

Parameters

<width>[1]	Query response
<NRf>	<NR3>
MINimum	2.0E–9
MAXimum	1.0E+0

[1] **The default for the parameter <delay_time> is seconds (S). You can also use the multipliers MS for milliseconds, US for microseconds, NS for nanoseconds, and PS for picoseconds.**

Reset Value

2.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set width to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:GLIT:WIDT 1.0`

Query: `TRIG:PULS:GLIT:WIDT?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs
TRIGger:PULSe:GLITch:QUALify

TRIGger:PULSe:SOURce
TRIGger:PULSe:SOURce?

Sets or queries the source of the trigger signal used for all pulse triggering. To activate pulse triggering, you must set TRIGger:TYPE to PULSe.

Setting source to INTernal1 selects the signal from INPut1 and [SENSe:]VOLTage1 to be the trigger source. The defined alias for the SCPI trigger :SEQuence[1] is A.

Syntax

```
TRIGger[:A]:PULSe:SOURce <source>
TRIGger[:A]:PULSe:SOURce?
```

Parameters

<source>	Query response
INTernal[1]	INT
INTernal2	INT2
INTernal3	INT3
INTernal4	INT4

Reset Value

INT

Errors and Events

Execution Error –224, “Illegal parameter value”
Attempted to set source to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:SOUR INT3`

Query: `TRIG:PULS:SOUR?`

Response: `INT3`

Related Commands

TRIGger:TYPE

TRIGger:PULSe:THReshold
TRIGger:PULSe:THReshold?

Sets or queries the voltage threshold used for pulse triggering. The threshold is the voltage level that the pulse trigger source must cross to mark the beginning and end of a pulse. To set positive or negative pulse polarity use the command TRIG:PULS:WIDT:POL or TRIG:GLIT:WIDT:POL, depending on the selected type of pulse triggering.

Syntax

```
TRIGger[:A]:PULSe:THReshold <level>
TRIGger[:A]:PULSe:THReshold?
```

Parameters

<level>[1]	Query response
<NRf>	<NR3>
Internal DC Coupled:	
absolute maximum[2]: –200.0 ≤ N ≤ 200.0 limited by ± VOLT:RANG:PTP steps 0.002 * VOLT:RANG:PTP	VOLT:RANG:OFFS ± VOLT:RANG:PTP
MINimum[2]	VOLT:RANG:OFFS – VOLT:RANG:PTP
MAXimum[2]	VOLT:RANG:OFFS + VOLT:RANG:PTP

[1] **The default for the parameter <level> is V for volts. You can also use the multipliers MV for millivolts and UV for microvolts.**

[2] **When you connect a probe, the maximum limits increase just as the vertical range increases for the input. For example, with a 1 V vertical range, connecting a 10X probe increases the vertical range and trigger level range to 10 V.**

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set threshold to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:THR 1.0`

Query: `TRIG:PULS:THR?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE

TRIGger:PULSe:TIMEout:POLarity TRIGger:PULSe:TIMEout:POLarity?

TVS600A Models Only

Sets or queries the polarity for pulse timeout triggering.

:A is a pre–defined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is set to PULSe and TRIGgger:PULSe:CLASs is set to TIMEout.

Syntax

```
TRIGger[:A]:PULSe:TIMEout:POLarity <polarity>
TRIGger[:A]:PULSe:TIMEout:POLarity?
```

Parameters

<polarity>	Query response
NEGative POSitive	NEG POS

Suffix Units and Multipliers

None

Reset Value

POS

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set polarity to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:TIME:POL NEG`

Query: `TRIG:PULS:TIME:POL?`

Response: `NEG`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs
TRIGger:PULSe:TIMEout:WIDTh

TRIGger:PULSe:TIMEout:WIDTh TRIGger:PULSe:TIMEout:WIDTh?

TVS600A Models Only

Sets or queries the width of the pulse.

:A is a pre–defined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is set to PULSe and TRIGger:PULSe:CLASs is set to TIMEout.

Syntax

```
TRIGger[:A]:PULSe:TIMEout:WIDTh <width>
TRIGger[:A]:PULSe:TIMEout:WIDTh?
```

Parameters

<width>	Query response
<NRf>	<NR3>
MINimum	2.0E–9
MAXimum	1.0E+0

Suffix Units and Multipliers

S (default), MS, US, NS, or PS

Reset Value

2.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set width to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:TIME:WIDT 1.0`

Query: `TRIG:PULS:TIME:WIDT?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs
TRIGger:PULSe:TIMEout:POLarity

TRIGger:PULSe:WIDTh:HLIMit
TRIGger:PULSe:WIDTh:HLIMit?

Sets or queries the high or longest valid pulse width for to qualify for pulse triggering. After a pulse on the trigger source begins, it cannot take longer than :WIDTh:HLIMit to cross the TRIGger:PULSe:THReshold level. HLIMit must be longer than :LLIMit. To activate pulse width triggering, set TRIGger:TYPE to PULSe and TRIGger:PULSe:CLASs to WIDTh.

Syntax

```
TRIGger[:A]:PULSe:WIDTh:HLIMit <hlimit>
TRIGger[:A]:PULSe:WIDTh:HLIMit?
```

Parameters

<hlimit>[1]	**Query response**
<NRf>	<NR3>
MIN MAX	2.0E–9 1.0E+0

[1] **The default for the parameter <hlimit> is seconds (S). You can also use the multipliers MS for milliseconds, US for microseconds, NS for nanoseconds, and PS for picoseconds.**

Reset Value

4.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set high limit to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:WIDT:HLIM 1.0`

Query: `TRIG:PULS:WIDT:HLIM?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs
TRIGger:PULSe:WIDTh:LLIMit

TRIGger:PULSe:WIDTh:LLIMit
TRIGger:PULSe:WIDTh:LLIMit?

Sets or queries the lower or minimum valid pulse width to qualify for pulse triggering. The pulse on the trigger source must take longer than the :WIDTh value to cross the TRIGger:PULSe:THReshold level. To activate this command, set TRIGger:TYPE to PULSe and TRIGger:PULSe:CLASs to WIDTh.

Syntax

```
TRIGger[:A]:PULSe:WIDTh:LLIMit <llimit>
TRIGger[:A]:PULSe:WIDTh:LLIMit?
```

Parameters

<llimit>[1]	Query response
<NRf>	<NR3>
MIN MAX	2.0E–9 1.0E+0

[1] **The default for the parameter <llimit> is seconds (S). You can also use the multipliers MS for milliseconds, US for microseconds, NS for nanoseconds, and PS for picoseconds.**

Reset Value

2.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set lower limit to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:WIDT:LLIM 0.1`

Query: `TRIG:PULS:WIDT:LLIM?`

Response: `100.0E-3`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs
TRIGger:PULSe:WIDTh:HLIMit

TRIGger:PULSe:WIDTh:POLarity
TRIGger:PULSe:WIDTh:POLarity?

Sets or queries the polarity of the pulse used for pulse width triggering. Use the command TRIGger:PULSe:WIDTh:QUALify to set whether the trigger event is for pulses inside or outside the limits (LLIMit and HLIMit).

This command is active only when you set TRIGger:TYPE to PULSe and TRIGger:PULSe:CLASs to WIDTh.

Syntax

```
TRIGger[:A]:PULSe:WIDTh:POLarity <polarity>
TRIGger[:A]:PULSe:WIDTh:POLarity?
```

Parameters

<polarity>	Query response
NEGative	NEG
POSitive	POS

Reset Value

POS

Errors and Events

Execution Error –224, “Illegal parameter value”
Attempted to set polarity to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:WIDT:POL NEG`

Query: `TRIG:PULS:WIDT:POL?`

Response: `NEG`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs

TRIGger:PULSe:WIDTh:QUALify
TRIGger:PULSe:WIDTh:QUALify?

Sets or queries the type of time qualification for pulse width triggering. The type IN triggers when the measured pulse width is within the specified lower and higher limits. OUT triggers when the pulse width is outside the specified lower and upper limits.

This command is active only when you set TRIGger:TYPE to PULSe and TRIGger:PULSe:CLASs to WIDTh.

Syntax

```
TRIGger[:A]:PULSe:WIDTh:QUALify <type>
TRIGger[:A]:PULSe:WIDTh:QUALify?
```

Parameters

<type>	Query response
IN OUT	IN OUT

Reset Value

IN

Errors and Events

Execution Error –224, "Illegal parameter value"
Attempted to set qualify to an illegal value.

Dependencies

None

Examples

Command: `TRIG:PULS:WIDT:QUAL OUT`

Query: `TRIG:PULS:WIDT:QUAL?`

Response: `OUT`

Related Commands

TRIGger:TYPE
TRIGger:PULSe:CLASs
TRIGger:PULSe:WIDTh:LLIMit
TRIGger:PULSe:WIDTh:HLIMit

TRIGger:SHOLdtime Subsystem (TVS600A Models Only)

This section describes each command and query in the TRIGger:SHOLdtime subsystem. See Figure 2–31. These commands are used to control setup and hold time violation triggering.

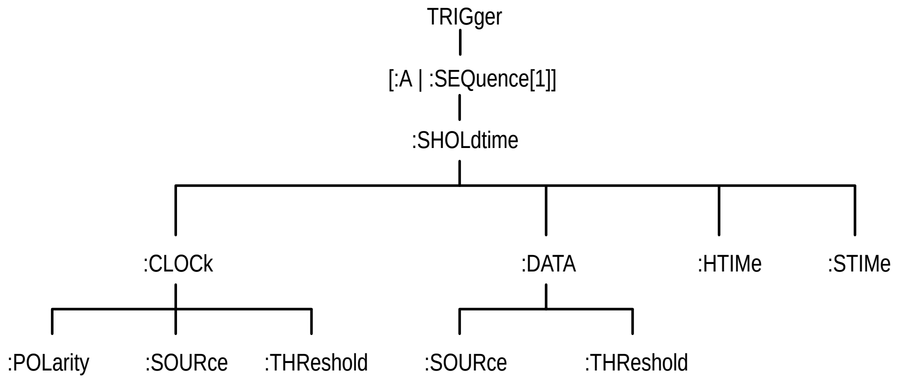


Figure 2–31: TRIGger:SHOLdtime subsystem hierarchy

TRIGger:SHOLdtime:CLOCk:POLarity TRIGger:SHOLdtime:CLOCk:POLarity?

TVS600A Models Only

Sets or queries which edge of the clock to use in checking setup and hold time violations.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is SHOLdtime.

Syntax

```
TRIGger[:A]:SHOLdtime:CLOCk:POLarity <polarity>
TRIGger[:A]:SHOLdtime:CLOCk:POLarity?
```

Parameters

<polarity>	Query response
NEGative	NEG
POSitive	POS

Suffix Units and Multipliers

None

Reset Value

POS

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set clock polarity to an illegal value.

Dependencies

None

Examples

Command: `TRIG:SHOL:CLOC:POL NEG`

Query: `TRIG:SHOL:CLOC:POL?`

Response: `NEG`

Related Commands

TRIGger:TYPE
TRIGger:SHOLdtime:CLOCk:SOURce

TRIGger:SHOLdtime:CLOCk:SOURce
TRIGger:SHOLdtime:CLOCk:SOURce?

TVS600A Models Only

Sets or queries the clock source channel.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is SHOLdtime.

Syntax

```
TRIGger[:A]:SHOLdtime:CLOCk:SOURce <source>
TRIGger[:A]:SHOLdtime:CLOCk:SOURce?
```

Parameters

<source>	Query response
INTernal1	INT1
INTernal2	INT2
INTernal3	INT3
INTernal4	INT4

Suffix Units and Multipliers

None

Reset Value

INT2

Errors and Events

Command Error –141, "Invalid character data"
Attempted to set clock source to an illegal value.

Dependencies

None

Examples

Command: `TRIG:SHOL:CLOC:SOUR INT3`

Query: `TRIG:SHOL:CLOC:SOUR?`

Response: `INT3`

Related Commands

TRIGger:TYPE
TRIGger:SHOLdtime:DATA:SOURce

TRIGger:SHOLdtime:CLOCk:THReshold TRIGger:SHOLdtime:CLOCk:THReshold?

TVS600A Models Only

Sets or queries the trigger threshold for the clock when setup and hold time violation triggering is selected.

:A is a predefined alias for :SEQuence[1].

The minimum and maximum values for threshold are VOLT:RANG:OFFS +/– VOLT:RANG:PTP (in VOLT:RANG:PTP * 0.002 steps).

This command only has effect when TRIGger:TYPE is SHOLdtime.

Syntax

```
TRIGger[:A]:SHOLdtime:CLOCk:THReshold <level>
TRIGger[:A]:SHOLdtime:CLOCk:THReshold?
```

Parameters

<level>	Query response
<NRf>	<NR3>
MINimum	<See Comments>
MAXimum	<See Comments>

Suffix Units and Multipliers

V (default), MV, or UV

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set threshold to an illegal value.

Dependencies

None

Examples

Command: `TRIG:SHOL:CLOC:THR 1.0`

Query: `TRIG:SHOL:CLOC:THR?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:SHOLdtime:CLOCk:SOURce

TRIGger:SHOLdtime:DATA:SOURce
TRIGger:SHOLdtime:DATA:SOURce?

TVS600A Models Only

Sets or queries the data source channel.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is SHOLdtime.

Syntax

```
TRIGger[:A]:SHOLdtime:DATA:SOURce <source>
TRIGger[:A]:SHOLdtime:DATA:SOURce?
```

Parameters

<source>	Query response
INTernal1	INT1
INTernal2	INT2
INTernal3	INT3
INTernal4	INT4

Suffix Units and Multipliers

None

Reset Value

INT1

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set data source to an illegal value.

Dependencies

None

Examples

Command: `TRIG:SHOL:DATA:SOUR INT3`

Query: `TRIG:SHOL:DATA:SOUR?`

Response: `INT3`

Related Commands

TRIGger:TYPE
TRIGger:SHOLdtime:CLOCk:SOURce

TRIGger:SHOLdtime:DATA:THReshold TRIGger:SHOLdtime:DATA:THReshold?

TVS600A Models Only

Sets or queries the trigger threshold for the data when setup and hold time violation triggering is selected.

:A is a predefined alias for :SEQuence[1].

The minimum and maximum values for threshold are VOLT:RANG:OFFS +/– VOLT:RANG:PTP (in VOLT:RANG:PTP * 0.002 steps).

This command only has effect when TRIGger:TYPE is SHOLdtime.

Syntax

```
TRIGger[:A]:SHOLdtime:DATA:THReshold <level>
TRIGger[:A]:SHOLdtime:DATA:THReshold?
```

Parameters

<level>	Query response
<NRf>	<NR3>
MINimum	<See comments>
MAXimum	<See comments>

Suffix Units and Multipliers

V (default), MV, or UV

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set threshold to an illegal value.

Dependencies

None

Examples

Command: `TRIG:SHOL:DATA:THR 1.0`

Query: `TRIG:SHOL:DATA:THR?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:SHOLdtime:DATA:SOURce

TRIGger:SHOLdtime:HTIMe TRIGger:SHOLdtime:HTIMe?

TVS600A Models Only

Sets or queries the hold time that violations will be checked against.

:A is a predefined alias for :SEQuence[1].

Positive values for hold time occur after the clock edge; negative values occure before the clock edge.

Keep hold time to at least 2ns less than the clock period or the trigger will not occur.

Positive values for both setup and hold times create a time zone that spans the clocking edge.

Negative setup and positive hold times create a time zone that follows the clocking edge.

Positive setup and negative hold times create a time violation zone that precedes the clocking edge.

The sum of the setup and hold times can not be less than 2ns. Setting one time will change the other time if this restriction is violated.

This command only has effect when TRIGger:TYPE is SHOLdtime.

Syntax

```
TRIGger[:A]:SHOLdtime:HTIMe <holdtime>
TRIGger[:A]:SHOLdtime:HTIMe?
```

Parameters

<holdtime>	Query response
<NRf>	<NR3>
MINimum MAXimum	–1.0E–9 102.0E–9

Suffix Units and Multipliers

S (default), MS, US, NS, or PS

Reset Value

2.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set hold time to an illegal value.

Dependencies	None
Examples	Command: `TRIG:SHOL:HTIM 1.0` Query: `TRIG:SHOL:HTIM?` Response: `1.0E+0`
Related Commands	TRIGger:TYPE TRIGger:SHOLdtime:CLOCk:POLarity

TRIGger:SHOLdtime:STIMe
TRIGger:SHOLdtime:STIMe?

TVS600A Models Only

Sets or queries the setup time that violations will be checked against.

:A is a predefined alias for :SEQuence[1].

Positive values for setup time occur before the clock edge; negative values occure after the clock edge.

Positive values for both setup and hold times create a time zone that spans the clocking edge.

Negative setup and positive hold times create a time zone that follows the clocking edge.

Positive setup and negative hold times create a time violation zone that precedes the clocking edge.

The sum of the setup and hold times can not be less than 2ns. Setting one time will change the other time if this restriction is violated.

This command only has effect when TRIGger:TYPE is SHOLdtime.

Syntax

```
TRIGger[:A]:SHOLdtime:STIMe <setuptime>
TRIGger[:A]:SHOLdtime:STIMe?
```

Parameters

<setup time>	Query response
<NRf>	<NR3>
MINimum MAXimum	–100.0E–9 100.0E–9

Suffix Units and Multipliers

S (default), MS, US, or PS

Reset Value

4.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set setup time to an illegal value.

Dependencies

None

Examples

Command: `TRIG:SHOL:STIM 1.0`

Query: `TRIG:SHOL:STIM?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:SHOLdtime:CLOCk:POLarity

TRIGger:TRANsition Subsystem

This section describes each command and query in the TRIGger:TRANsition subsystem. See Figure 2–32. These commands are used to control runt, slewrate, and combination triggering.

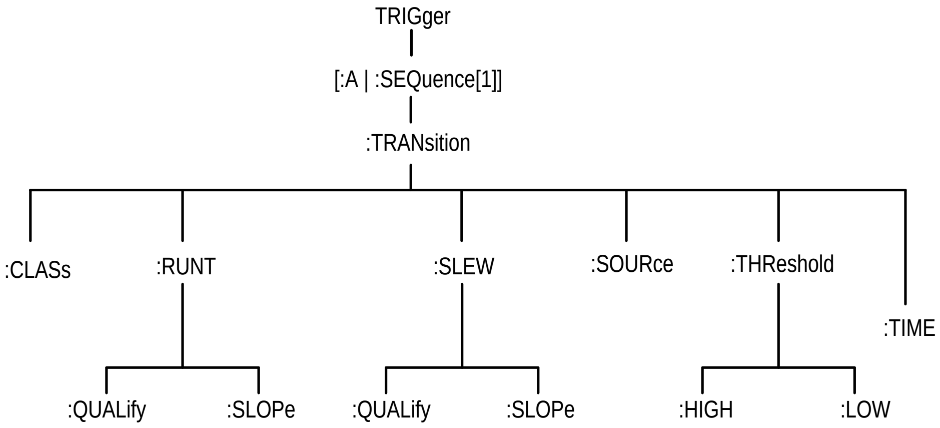


Figure 2–32: TRIGger:TRANsition subsystem hierarchy

TRIGger:TRANsition:CLASs
TRIGger:TRANsition:CLASs?

TVS600A Models Only

Sets or queries the class of transition triggering selected. RUNT: the instrument triggers when a pulse crosses one of two thresholds but does not cross the second threshold before recrossing the first. SLEWrate: the instrument triggers when an edge crosses two thresholds. All classes may be time qualified.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is TRANsition.

Syntax

```
TRIGger[:A]:TRANsition:CLASs <transition_class>
TRIGger[:A]:TRANsition:CLASs?
```

Parameters

<transition_class>	Query response
RUNT	RUNT
SLEWrate	SLEW

Suffix Units and Multipliers

None

Reset Value

RUNT

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set class to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:CLAS SLEWRATE`

Query: `TRIG:TRAN:CLAS?`

Response: `SLEW`

Related Commands

TRIGger:TYPE

TRIGger:TRANsition:RUNT:QUALify
TRIGger:TRANsition:RUNT:QUALify?

TVS600A Models Only

Sets or queries the type of time qualification for RUNT transition triggering. OFF means there is no time qualification. GT means the trigger occurs when the transition time is greater than the specified time.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is TRANsition, and TRIGger:TRANsition:CLASs is RUNT.

Syntax

```
TRIGger[:A]:TRANsition:RUNT:QUALify <type>
TRIGger[:A]:TRANsition:RUNT:QUALify?
```

Parameters

<type>	Query response
GT OFF	GT OFF

Suffix Units and Multipliers

None

Reset Value

OFF

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set QUALify to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:RUNT:QUAL GT`

Query: `TRIG:TRAN:RUNT:QUAL?`

Response: `GT`

Related Commands

TRIGger:TYPE
TRIGger:TRANsition:CLASs
TRIGger:TRANsition:TIME

TRIGger:TRANsition:RUNT:SLOPe TRIGger:TRANsition:RUNT:SLOPe?

TVS600A Models Only

Sets or queries the slope for RUNT transition triggering class.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is TRANsition, and TRIGger:TRANsition:CLASs is RUNT.

Syntax

```
TRIGger[:A]:TRANsition:RUNT:SLOPe <slope>
TRIGger[:A]:TRANsition:RUNT:SLOPe?
```

Parameters

<slope>	Query response
EITHer	EITH
NEGative	NEG
POSitive	POS

Suffix Units and Multipliers

None

Reset Value

POS

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set slope to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:RUNT:SLOP NEG`

Query: `TRIG:TRAN:RUNT:SLOP?`

Response: `NEG`

Related Commands

TRIGger:TYPE
TRIGger:TRANsition:CLASs

TRIGger:TRANsition:SLEW:QUALify TRIGger:TRANsition:SLEW:QUALify?

TVS600A Models Only

Sets or queries the type of time qualification for SLEW transition triggering. LT means the trigger occurs when the transition time is less than the specified time. GT means the trigger occurs when the transition time is greater than the specified time.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is TRANsition, and TRIGger:TRANsition:CLASs is SLEW.

Syntax

```
TRIGger[:A]:TRANsition:SLEW:QUALify <type>
TRIGger[:A]:TRANsition:SLEW:QUALify?
```

Parameters

<type>	Query response
GT LT	GT LT

Suffix Units and Multipliers

None

Reset Value

LT

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set qualify to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:SLEW:QUAL GT`

Query: `TRIG:TRAN:SLEW:QUAL?`

Response: `GT`

Related Commands

TRIGger:TYPE
TRIGger:TRANsition:CLASs
TRIGger:TRANsition:TIME

TRIGger:TRANsition:SLEW:SLOPe
TRIGger:TRANsition:SLEW:SLOPe?

TVS600A Models Only

Sets or queries the slope for SLEW transition triggering class.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is TRANsition, and TRIGger:TRANsition:CLASs is SLEW.

Syntax

```
TRIGger[:A]:TRANsition:SLEW:SLOPe <slope>
TRIGger[:A]:TRANsition:SLEW:SLOPe?
```

Parameters

<slope>	Query response
NEGative	NEG
POSitive	POS

Suffix Units and Multipliers

None

Reset Value

POS

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set slope to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:SLEW:SLOP NEG`

Query: `TRIG:TRAN:SLEW:SLOP?`

Response: `NEG`

Related Commands

TRIGger:TYPE
TRIGger:TRANsition:CLASs

TRIGger:TRANsition:SOURce
TRIGger:TRANsition:SOURce?

TVS600A Models Only

Sets or queries the trigger source for all transition triggering classes.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is TRANsition.

Setting source to INTernal1 causes the signal from INPut1 and [SENSe:]VOLTage1 to be the trigger source.

Syntax

```
TRIGger[:A]:TRANsition:SOURce <source>
TRIGger[:A]:TRANsition:SOURce?
```

Parameters

<source>	Query response
INTernal	INT
INTernal1	INT1
INTernal2	INT2
INTernal3	INT3
INTernal4	INT4

Suffix Units and Multipliers

None

Reset Value

INT

Errors and Events

Command Error –141, “Invalid character data”
Attempted to set source to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:SOUR INT3`

Query: `TRIG:TRAN:SOUR?`

Response: `INT3`

Related Commands

TRIGger:TYPE

TRIGger:TRANsition:THReshold:HIGH TRIGger:TRANsition:THReshold:HIGH?

TVS600A Models Only

Sets or queries the high threshold for transition triggering.

:A is a predefined alias for :SEQuence[1].

The minimum and maximum values for threshold are VOLT:RANG:OFFS +/– VOLT:RANG:PTP (in VOLT:RANG:PTP * 0.002 steps).

This command only has effect when TRIGger:TYPE is TRANsition.

Syntax

```
TRIGger[:A]:TRANsition:THReshold:HIGH <level>
TRIGger[:A]:TRANsition:THReshold:HIGH?
```

Parameters

<level>	Query response
<NRf>	<NR3>
MINimum	<See comments>
MAXimum	<See comments>

Suffix Units and Multipliers

V (default), MV, or UV

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set threshold to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:THR:HIGH 1.0`

Query: `TRIG:TRAN:THR:HIGH?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:TRANsition:THReshold:LOW

TRIGger:TRANsition:THReshold:LOW
TRIGger:TRANsition:THReshold:LOW?

TVS600A Models Only

Sets or queries the low threshold for transition triggering.

:A is a predefined alias for :SEQuence[1].

The minimum and maximum values for threshold are VOLT:RANG:OFFS +/– VOLT:RANG:PTP (in VOLT:RANG:PTP * 0.002 steps).

This command only has effect when TRIGger:TYPE is TRANsition.

Syntax

```
TRIGger[:A]:TRANsition:THReshold:LOW <level>
TRIGger[:A]:TRANsition:THReshold:LOW?
```

Parameters

<level>	Query response
<NRf>	<NR3>
MINimum MAXimum	<See comments> <See comments>

Suffix Units and Multipliers

V (default), MV, UV

Reset Value

0.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set threshold to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:THR:LOW -1.0`

Query: `TRIG:TRAN:THR:LOW?`

Response: `-1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:TRANsition:THReshold:HIGH

TRIGger:TRANsition:TIME
TRIGger:TRANsition:TIME?

TVS600A Models Only

Sets or queries the transition qualification time.

:A is a predefined alias for :SEQuence[1].

This command only has effect when TRIGger:TYPE is TRANsition and TRIGger:TRANsition:QUALify is LT or GT.

Syntax

```
TRIGger[:A]:TRANsition:TIME <time>
TRIGger[:A]:TRANsition:TIME?
```

Parameters

<time>	Query response
<NRf>	<NR3>
MINimum	2.0E–9
MAXimum	1.0E+0

Suffix Units and Multipliers

S (default), MS, US, or PS

Reset Value

2.0E–9

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set time to an illegal value.

Dependencies

None

Examples

Command: `TRIG:TRAN:TIME 1.0`

Query: `TRIG:TRAN:TIME?`

Response: `1.0E+0`

Related Commands

TRIGger:TYPE
TRIGger:TRANsition:QUALify

VOLTage Subsystem

This section describes each command and query in the [SENSe:]VOLTage subsystem. Figure 2–33 shows the command tree for the VOLTage subsystem. Figure 2–34 shows the part of the waveform analyzer controlled by the VOLTage commands. The input channel number, defined at the probe connector, is shared as the parameter <n> for the INPut<n> and VOLTage<n> commands. Only the four channel TVS641A and TVS645A will accept VOLTage3 and VOLTage4 commands.

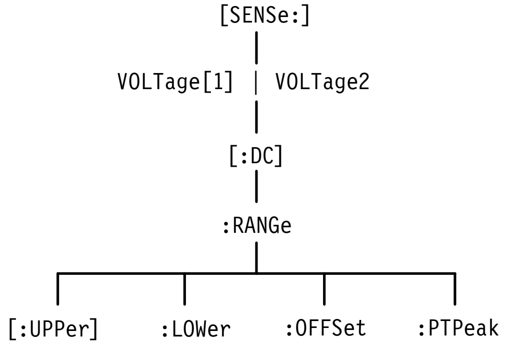


Figure 2–33: VOLTage subsystem hierarchy

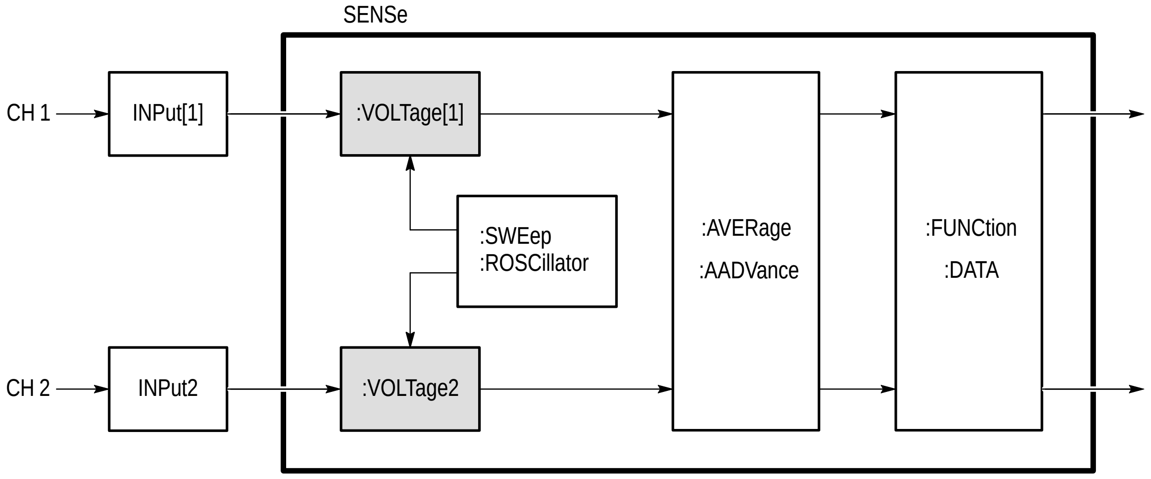


Figure 2–34: VOLTage subsystem functional model

VOLTage:RANGe[:UPPer]
VOLTage:RANGe[:UPPer]?

Sets or queries the most positive end of the amplifier voltage range. You should set the amplifier voltage range to match the signal amplitude you expect to acquire with the digitizer. The string :UPPer may be omitted as shown in the examples that follow.

The range of permitted values for <upper> depends on the attached probe. Attached probes modify the full-scale range by the attenuation factor of the probe. For example, a 10X probe will cause the 100 mV range to become a 1 V range. This in turn causes the minimum and maximum offset values to change from ±1 V to ±10 V.

Syntax

```
[SENSe:]VOLTage<n>[:DC]:RANGe[:UPPer] <upper>
[SENSe:]VOLTage<n>[:DC]:RANGe[:UPPer]?
```

Parameters

<n> (Channel number)[1]	Query response
1 2 3 4	Not applicable

[1] The input channel number <n> is used by the commands INPut<n> and VOLTage<n>. If you omit <n>, the default is channel 1.

<upper> (Upper limit of voltage range)[1]	Query response
<NRf>	<NR3>

[1] The default multiplier for the <upper> parameter is V for volts. You can also use MV for millivolts.

Reset Value

0.5E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set upper to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to program VOLTage3 or VOLTage4 on an instrument configured with only two channels.

Dependencies

VOLTage:RANGe[:UPPer] and VOLTage:RANGe:LOWer operate as a pair. Changing one will not cause the other to change. However, a change in the voltage range will change VOLTage:RANGe:PTPeak and VOLTage:RANGe:OFFSet so they describe the new voltage range.

VOLTage:RANGe:LOWer is not returned in the response to *LRN? because the voltage range is returned as VOLT:RANG:PTP and VOLT:RANG:OFFS.

Examples

Command: `VOLT1:RANG 2`

Query: `VOLT1:RANG?`

Response: `2.0E+0`

Related Commands

VOLTage:RANGe:LOWer
VOLTage:RANGe:OFFSet
VOLTage:RANGe:PTPeak

VOLTage:RANGe:LOWer
VOLTage:RANGe:LOWer?

Sets or queries the most negative end of the amplifier voltage range. You should set the amplifier voltage range to match the signal amplitude you expect to acquire with the digitizer.

The range of permitted values for <lower> depends on the attached probe. An attached probe modifies the full-scale range by the attenuation factor of the probe. For example, a 10X probe causes the 100 mV range to become a 1 V range and the minimum and maximum offset values to change from ±1 V to ±10 V.

Syntax

```
[SENSe:]VOLTage<n>[:DC]:RANGe:LOWer <lower>
[SENSe:]VOLTage<n>[:DC]:RANGe:LOWer?
```

Parameters

<n> (Channel number)[1]	Query response
1 2 3 4	Not applicable

[1] **The input channel number <n> is used by the commands INPut<n> and VOLTage<n>. If you omit <n>, the default is channel 1.**

<lower> (Lower limit of voltage range)[1]	Query response
<NRf>	<NR3>

[1] The default multiplier for the <lower> parameter is V for volts. You can also use MV for millivolts.

Reset Value

–0.5E+0

Errors and Events

Execution Error –222, "Data out of range"
Attempted to set low pass frequency to an illegal value.

Execution Error –241, "Hardware missing"
Attempted to program VOLTage3 or VOLTage4 on an instrument configured with only two channels.

Dependencies

VOLTage:RANGe[:UPPer] and VOLTage:RANGe:LOWer operate as a pair. Changing one will not cause the other to change. However, a change in the voltage range will change VOLTage:RANGe:PTPeak and VOLTage:RANGe:OFFSet so they describe the new voltage range.

VOLTage:RANGe:LOWer is not returned in the response to *LRN? because the voltage range is returned as VOLT:RANG:PTP and VOLT:RANG:OFFS.

Examples

Command: `VOLT1:RANG:LOW −2`

Query: `VOLT1:RANG:LOW?`

Response: `−2.0E+0`

Related Commands

VOLTage:RANGe[:UPPer]
VOLTage:RANGe:OFFSet
VOLTage:RANGe:PTPeak

VOLTage:RANGe:OFFSet
VOLTage:RANGe:OFFSet?

Sets or queries the voltage offset of the specified input amplifier. The offset value is subtracted from the input signal which allows you to move the signal up or down into the amplifier vertical range. A positive value removes positive DC offset from the signal and a negative offset value removes negative DC offset. A common use of offset is to subtract the DC component of a signal so you can acquire the AC portion at higher vertical resolution. Measurement results do not change when you change offset.

The range of permitted values and the smallest increment for offset depend on the setting of the peak-to-peak range (VOLTage:RANGe:PTPeak) and on the attached probe. An attached probe modifies the full-scale range by the attenuation factor of the probe. For example, a 10X probe causes the 100 mV range to become a 1 V range and the minimum and maximum offset values to change from ±1 V to ±10 V.

Syntax

```
[SENSe:]VOLTage<n>[:DC]:RANGe:OFFSet <offset>
[SENSe:]VOLTage<n>[:DC]:RANGe:OFFSet?
```

Parameters

<n> (Channel number)[1]	Query response
1 2 3 4	Not applicable

[1] **The input channel number <n> is used by the commands INPut<n> and VOLTage<n>. If you omit <n>, the default is channel 1.**

<offset> (Value subtracted from signal)[1]	Query response
<NRf>	<NR3>
PTPeak 10 mV – 1 V ±1.0 (in 1 mV steps) MINimum MAXimum	 –1.0E+0 1.0E+0
PTPeak 1.01 V – 10 V ±10.0 (in 10 mV steps) MINimum MAXimum	 –10.0E+0 10.0E+0

<offset> (Value subtracted from signal)[1]	Query response
PTPeak 10.1 V – 100 V	
±100.0 (in 100 mV steps)	
MINimum	–100.0E+0
MAXimum	100.0E+0

[1] **The default multiplier for the <offset> parameter is V for volts. You can also use MV for millivolts.**

Reset Value

0.0E+0

Errors and Events

Execution Error –222, "Data out of range"
Attempted to set offset to an illegal value.

Execution Error –241, "Hardware missing"
Attempted to program VOLTage3 or VOLTage4
on an instrument configured with two channels.

Dependencies

VOLTage:RANGe:PTPeak and VOLTage:RANGe:OFFSet operate as a pair. Changing the voltage range, changes VOLTage:RANGe[:UPPer] and VOLTage:RANGe:LOWer so they describe the new voltage range.

Examples

Command: `VOLT1:RANG:OFFS 1`

Query: `VOLT1:RANG:OFFS?`

Response: `1.0E+0`

Related Commands

VOLTage:RANGe:PTPeak
VOLTage:RANGe:LOWer
VOLTage:RANGe[:UPPer]

VOLTage:RANGe:PTPeak
VOLTage:RANGe:PTPeak?

Sets or queries the peak-to-peak (full-scale) voltage range of the specified input amplifier. The range of permitted values and the smallest increment for offset depend on the setting of the peak-to-peak range (VOLTage:RANGe:PTPeak). For specific values, see the listing of <range> values under *Parameters*.

The values permitted for <range> also depend on the attached probe. An attached probe modifies the full-scale range by the attenuation factor of the probe. For example, a 10X probe changes the 10 mV range to the 100 mV range.

Syntax

```
[SENSe:]VOLTage<n>[:DC]:RANGe:PTPeak <range>
[SENSe:]VOLTage<n>[:DC]:RANGe:PTPeak?
```

Parameters

<n> (Channel number)[1]	Query response
1 2 3 4	Not applicable

[1] **The input channel number <n> is used by the commands INPut<n> and VOLTage<n>. If you omit <n>, the default is channel 1.**

<range> (Range for vertical amplifier)[1]	Query response
<NRf>	<NR3>
10 mV – 20 mV (100 µV steps) 20 mV – 50 mV (200 µV steps) 50 mV – 100 mV (500 µV steps) 100 mV – 200 mV (1 mV steps) 200 mV – 500 mV (2 mV steps) 500 mV – 1 V (5 mV steps) 1 V – 2 V (10 mV steps) 2 V – 5 V (20 mV steps) 5 V – 10 V (50 mV steps) 10 V – 20 V (100 mV steps) 20 V – 50 V (200 mV steps) 50 V – 100 V (500 mV steps)	
MINimum MAXimum	10.0E–3 100.0E+0

[1] **The default multiplier for the <offset> parameter is V for volts. You can also use MV for millivolts.**

Reset Value

1.0E+0

Errors and Events

Execution Error –222, “Data out of range”
Attempted to set range to an illegal value.

Execution Error –241, “Hardware missing”
Attempted to program VOLTage3 or VOLTage4
on an instrument configured with two channels.

Dependencies

VOLTage:RANGe:PTPeak and VOLTage:RANGe:OFFSet together define what part of the input signal to acquire. Changing the voltage range with VOLTage:RANGe:PTPeak does change VOLTage:RANGe[:UPPer] and VOLTage:RANGe:LOWer, because together they provide an alternate way to query, or define, the voltage range for acquisition.

Examples

Command: `VOLT1:RANG:PTP 100`

Query: `VOLT1:RANG:PTP?`

Response: `100.0E+0`

Related Commands

VOLTage:RANGe:OFFSet
VOLTage:RANGe:[UPPer]
VOLTage:RANGe:LOWer

IEEE 488.2 Common Commands

This section describes the IEEE 488.2 common commands. Figure 2–35 shows the common syntax for these commands. The program mnemonics are described in alphabetical order in this section.

```
*<program mnemonic>[?]
```

Figure 2–35: IEEE 488.2 Common Command Syntax

*CAL? (Query Only)

Initiates internal calibration and returns a failure code. If more than one failure occurs, the response includes only the number of the first failure. You can obtain additional test results with the command CALibration:RESult:VERBose?.

*CAL? and CALibration? perform the same function.

Syntax

```
*CAL?
```

Parameters

<failure_code>	Query response
Not applicable	<NR1> 0 (No failures) 2000 to 2999 (Calibration failure)

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `*CAL?`

Response: `0`

Related Commands

CALibration
CALibration:RESults:VERBose?

*CLS

Clears, or nulls, the SCPI and IEEE 488.2 event registers and the Status Queue. However, the SCPI and IEEE 488.2 enable registers are not cleared. For more information on the Status Queue and the event registers, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
*CLS
```

Parameters

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `*CLS`

Related Commands

STATus:PRESet

*ESE
*ESE?

Sets or queries the Event Status Enable Register (ESER). The ESER determines which events in the Standard Event Status Register can set the Event Status bit (bit 5) in the Status Byte Register. The bits in the ESER correspond to the bits in the Standard Event Status Register, which are defined in Table 2–6 on page 2–315.

The STATus:PRESet command does not change the Event Status Enable Register. For more information on the Status and Events reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
*ESE <mask>
*ESE?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal number> 0 ≤ N ≤ #HFF	<NR1>

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `*ESE #H40`
This command enables only the User Request bit (bit6) in the Standard Event Status Register. No other events can set the Event Status bit in the Status Byte Register.

Query: `*ESE?`

Response: `64`

Related Commands

*ESR?

*ESR? (Query Only)

Returns the contents of the Standard Event Status Register as a decimal number. More than one event may be reported at the same time. The Status Queue contains a chronological record of the events. Table 2–6 describes the bit assignments in the Standard Event Status Register. Reading this register clears it. For more information on the Status and Events reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Table 2–6: The Standard Event Status Register

Bit	Decimal Value	Function
0	1	**Operation Complete** shows that the operation is complete. This bit is set when all pending operations complete following an *OPC command.
1	2	**Request Control** (not used)
2	4	**Query Error** shows that the waveform analyzer attempted to read the Output Queue when no data was present or pending, or that data in the Output Queue was lost.
3	8	**Device Dependent Error** shows that a device error occurred. Refer to *Status and Events* in the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for device error messages.
4	16	**Execution Error** shows that an error occurred while the waveform analyzer was executing a command or query. Refer to *Status and Events* in the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for execution error messages.
5	32	**Command Error** shows that an error occurred while the waveform analyzer was parsing a command or query. Refer to *Status and Events* in the *TVS600 & TVS600A Series Waveform Analyzers User Manual* for command error messages.
6	64	**User Request** indicates that a probe ID button was pressed.
7	128	**Power On** shows that the waveform analyzer was powered on.

Syntax

```
*ESR?
```

Parameters

<event>	Query response
Not applicable	<NR1>

Reset Value

Not applicable

Errors and Events

None

Dependencies Reading this register clears it.

Examples Query: `*ESR?`

Response: `40`

Related Commands *ESE

*IDN? (Query Only)

Returns the waveform analyzer identification message. The fourth field of the identification message includes both the SCPI and firmware version numbers.

Syntax `*IDN?`

Parameters

<id_message>	Query response
Not applicable	Manufacturer, model number, serial number, SCPI and firmware version numbers (syntax defined by IEEE 488.2)

Reset Value Not applicable

Errors and Events None

Dependencies None

Examples Query: `*IDN?`

Response: `TEKTRONIX,TVS641A,B020100,SCPI:95.0 FVER:2.0.25`

Related Commands SYStem:VERSion?

*LRN? (Query Only)

Returns the current state of the waveform analyzer as a sequence of ASCII settings. You can store and return these settings as a group to the waveform analyzer to place it in a known state. Command headers and parameters in the returned settings are in the short form.

Syntax

```
*LRN?
```

Parameters

<ascii_setting>	Query response
Not applicable	<Program Message Unit> {,<Program Message Unit>}

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Query: `*LRN?`

Response:

```
INP1:COUP DC;FILT 1;
FILT:FREQ 20.0E+6;
IMP 50.0E+0;:INP2 ....
```

Related Commands

SYSTem:SET
*RCL
*SAV

*OPC
*OPC?

Synchronizes command execution with the controller. *OPC causes an Operation Complete event to be generated when the active command finishes. *OPC? causes an ASCII “1” to be placed in the output queue when the command completes.

Syntax

```
*OPC
*OPC?
```

Parameters

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command:	`*OPC`
Query:	`*OPC?`
Response:	`1`

Related Commands

*WAI

*OPT? (Query Only)

Returns the options installed in the instrument as a comma-separated list. A zero response indicates no option is installed.

Syntax

```
*OPT?
```

Parameters

<options>	Query response
Not applicable	<arbitrary ascii response data>

Reset Value Not applicable

Errors and Events None

Dependencies None

Examples

Query: `*OPT?`

Response: 0

Related Commands *IDN?

*PUD
*PUD?

Sets or queries the protected user data stored in the waveform analyzer. The user data can not exceed 1023 characters. You can use *PUD to provide a corporate identification or name for your waveform analyzer. You must set SYSTem:PROTect to OFF before attempting to modify the user data with *PUD. The query form may be used at any time.

Syntax

```
*PUD <data>
*PUD?
```

Parameters

<data>	Query response
<string> (up to 1023 characters)	<string>

Reset Value

Not applicable

Errors and Events

Execution Error –203, “Command protected”
Attempted to set PUD without first removing the
restriction with the SYSTem:PROTect command.

Execution Error –223, “Too much data”
Attempted to set PUD to a string longer than 1023 characters.

Dependencies

None

Examples

Command: `*PUD "WA 4-1"`

Query: `*PUD?`

Response: `"WA 4-1"`

Related Commands

SYSTem:PROTect

*RCL

Recalls the specified instrument setting from one of the ten non-volatile memory locations in the waveform analyzer. You save settings to these locations with the *SAV command.

Syntax

```
*RCL <location>
```

Parameters

<location>	Query response
<NRf> 1 to 10	Not applicable

Reset Value

Not applicable

Errors and Events

Execution Error –221, “Settings conflict”
Attempted to recall an instrument setting from an empty memory location.

Execution Error –224, “Illegal parameter value”
Attempted to recall an instrument setting from an illegal memory location.

Dependencies

None

Examples

Command: `*RCL 4`

Related Commands

SYSTem:SET
*LRN?
*SAV

*RST

Resets instrument settings to a default state. Each command description includes a reset value if appropriate. Many settings in the SYSTem and STATus subsystems are not reset by this command.

Syntax

```
*RST
```

Parameters

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `*RST`

Related Commands

STATus:PRESet
*CLS

*SAV

Saves the current instrument settings in one of the ten nonvolatile memory locations in the waveform analyzer. You can recall these settings at a later time with the *RCL command.

Syntax

```
*SAV <location>
```

Parameters

<location>	Query response
<NRf> 1 to10	Not applicable

Reset Value

Not applicable

Errors and Events

Execution Error -224, “Illegal parameter value”
Attempted to save instrument settings to an illegal memory location.

Dependencies

None

Examples

Command: `*SAV 4`

Related Commands

SYSTem:SET
*LRN?
*RCL

*SRE
*SRE?

Sets or queries the Service Request Enable Register (SRER). The SRER determines which events in the Status Byte Register can generate an interrupt to the system controller. The SRER bits correspond to the bits in the Status Byte Register, which are described in Table 2–7 on page 2–325.

STATus:PRESet does not change the Service Request Enable Register. For more information on the Status and Events reporting system, refer to the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Syntax

```
*SRE <mask>
*SRE?
```

Parameters

<mask>	Query response
<NRf> | <Non-decimal numeric> 0 ≤ N ≤ #HFF	<NR1>

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `*SRE #H60`

Query: `*SRE?`

Response: `96`

Related Commands

*STB?

*STB? (Query Only)

Returns the contents of the Status Byte Register as a decimal sum of all set events. You can get a chronological list of events from the Status Queue using the SYSTem:ERRor? query. Table 2–7 describes the bit assignments in the Status Byte Register. For a complete description of the status and events reporting system, see the *TVS600 & TVS600A Series Waveform Analyzers User Manual*.

Table 2–7: The Status Byte Register

Bit	Decimal Value	Function
0–1	–	Not used.
2	4	**Error/Event Queue not empty** indicates that the error/event queue contains information and is waiting to be read.
3	8	**Questionable Event Status** indicates that the quality of result data or of an operation is questionable.
4	16	**Message Available (MAV)** shows that data is available in the Output Queue.
5	32	**Event Status Bit** indicates that one or more events have occurred in the Standard Event Status Register.
6	64	**Master Summary Status (MSS)** is a summary bit that indicates other bits in the Status Byte Register are set.
7 (MSB)	128	**Operation Event Register** indicates that the waveform analyzer is busy performing a normal operation such as acquiring a waveform.

Syntax

```
*STB?
```

Parameters

<event>	Query response
Not applicable	<NR1>

Reset Value

Not applicable

Errors and Events

None

Dependencies

Reading this register does not clear it.

Examples

Query: `*STB?`

Response: `96`
This response indicates the MAV and the MSS bits are set.

Related Commands

*SRE

*TRG

Arms the waveform analyzer to accept triggers from the VXIbus. Before using *TRG, you must start the Arm/Trigger system with the INITiate command and use ARM:SOURce to select BUS.

Syntax

```
*TRG
```

Parameters

None

Reset Value

Not applicable

Errors and Events

Execution Error –212, “Arm ignored”
Sent the *TRG command or the VXIbus word serial command <Trigger> when ARM:SOURce was not set to BUS or when the instrument was not in the “Waiting for Arm” state.

Dependencies

None

Examples

Command: `*TRG`

Related Commands

INITIATE
ARM:SOURCE

*TST? (Query Only)

Initiates an internal self-test and returns a failure code. If more than one failure occurred, only the test number of the first failure is reported. A value of zero indicates there were no failures. This query is only valid when looping is disabled. Detailed test results are available with the TEST:RESult:VERBose? command.

*TST? and TEST? perform the same function.

Syntax

```
*TST?
```

Parameters

<failure_code>	Query response
Not applicable	<NR1> 0 (no failure) 1000 – 1999 (Self-test failure) 2000 – 2999 (Calibration failure)

Reset Value

Not applicable

Errors and Events

Execution Error –221, “Settings conflict”
Executed *TST? when TEST:CONTrol:LOOP is ON.

Dependencies

None

Examples

Query: `*TST?`

Response: 0

Related Commands

TEST
TEST:RESults:VERBose?

*WAI

Synchronizes command execution with the system controller. *WAI prevents the waveform analyzer from executing further commands until the active command completes execution.

Syntax

```
*WAI
```

Parameters

None

Reset Value

Not applicable

Errors and Events

None

Dependencies

None

Examples

Command: `*WAI`

Related Commands

*OPC

Appendices

Appendix A: Expression Syntax

Expression Syntax

The syntax for CALCulate expressions is defined in the following BNF description (for meanings of BNF forms, see Table 1–2 *BNF Symbols and Meanings* on page 1–7). Note that TVS600 models cannot use elements that refer to TVS600A features only.

```
<statement>    ::=    expr“;”
                      <statement><statement>
                       /* epsilon */
<expr>         ::=    <statement><expr>
                      <factor>
                      <expr><binop><expr>
                      <expr><logop><expr>
                      <expr><relop><expr>
                      <lval><asgnop><expr>
                      <lval>“#=”<accumop>“(”<arg_list>“)”
                      <lval>“:=”<expr>
                      <expr>“..”<expr>
                      <expr>{“<?”“|”“>?”}<expr>
                      !<expr>
                      “{”<statement>“}”
                      /* epsilon */
factor         ::=    [–<number>|{+}<number>]{<unit>}
                      AATS
                      [CHAN1|CHAN2|CHAN3|CHAN4]{“[”<nrx>“]”}
                      <meas>
                      <parameter>
                      <ref>
                      “(”<expr>“)”
<lval>         ::=    <ref>|<parameter>
<meas>         ::=    <func>“(”<arg_list>“)”
                      <meas_func>“(”<m_arg_list>“)”
<arg_list>     ::=    <expr>{[, <expr>]...}
<m_arg_list>   ::=    <wmp_expr> {[, wmp_expr>]...}
<wmp_expr>     ::=    {WMP1:|WMP2:|WMP3:|WMP4:} <expr>
<parameter>    ::=    %1|%2|%3|%4|%5|%6|%7|%8|%9
<ref>          ::=    REF1|REF2|REF3|REF4|REF5|REF6|REF7|REF8|REF9|REF10|REF
                      “[”<nrx>“]”
<binop>        ::=    +|–|*|/
<asgnop>       ::=    +=|–=|*=|/=||=|&=|^=
<logop>        ::=    &|||^|AND|OR
<relop>        ::=    ==|!=|>|>=|<|<=|><|<>|EQ|NE|GT|GE|LT|LE|INSide|OUTside
<accumop>      ::=    AVERage|ENVelope|VECTor|STATistics
```

```
<meas_func>   ::=   AAMList|AC|ACRMS|AMPLitude|AREA|DC|DELay
                    |CARea|CMEan|COPulse|CPARea|CRMS|CROss
                    ||FTIMe|FREQuency|GAIN|HIGH|LOW|MAXimum
                    |MEAN|MID|MINimum|NCRoss|NDUTycycle|NWIDth
                    |OVERshoot|PARea|PCRoss|PDUTycycle|PERiod|PHASe
                    |PREShoot|PTPeak|PWIDth|RMS|RTIMe|SDEViation|TTRig|WMList
<func>        ::=   ABSolute|AVERage|BAT|DERivative|ENVelope|EVT|FFT|
                    FILTer|FORMat|HLT|INTegral|SEGMent|SMOothing|SRQ|
                    STATistics|TRANsform|TRG|VECTor|XDURation|XOFFset|XSCale|
                    XSIZe
<nrx>         ::=   <number>{<unit>}
<unit>        ::=   PCT|XUnit|Second|PS|NS|US|MS
<number>      ::=   [0–9]...{.[0–9]...}{[E|e]{+|–}[0–9]...}
```

An example of the expression syntax follows:

```
CALC1:PATH:EXPR (FORM(TRAN(FILT(CHAN1))))
```

This command performs frequency filtering on CHAN1, then performs an FFT transform, and finally formats the result, perhaps in a logarithmic magnitude format.

The settings for `CALC:FILTer`, `:TRANsform`, and `:FORMat` functions must be completed in separate command statements prior to starting acquisition. The settings might include enabling the low-pass filter with `CALC:FILT:FREQ:LPAS`, setting the TRANsform WINDow type, and setting the result format for the resulting FFT waveform record with `CALC:FORM MLOG`.

The following example produces a waveform with an average value of zero and an amplitude normalized to one:

```
CALC2:PATH:EXPR ((CHAN1–MEAN(CHAN1))/AMPL(CHAN1))
```

Appendix B: ASCII Character Chart

Table B–1: ASCII Code Chart

B7 B6 B5 BITS B4 B3 B2 B1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		NUMBERS SYMBOLS		UPPER CASE		LOWER CASE	
0 0 0 0	0 NUL 0 0	20 DLE 10 16	40 SP 20 32	60 0 30 48	100 @ 40 64	120 P 50 80	140 \` 60 96	160 p 70 112
0 0 0 1	1 SOH 1 1	21 DC1 11 17	41 ! 21 33	61 1 31 49	101 A 41 65	121 Q 51 81	141 a 61 97	161 q 71 113
0 0 1 0	2 STX 2 2	22 DC2 12 18	42 " 22 34	62 2 32 50	102 B 42 66	122 R 52 82	142 b 62 98	162 r 72 114
0 0 1 1	3 ETX 3 3	23 DC3 13 19	43 # 23 35	63 3 33 51	103 C 43 67	123 S 53 83	143 c 63 99	163 s 73 115
0 1 0 0	4 EOT 4 4	24 DC4 14 20	44 $ 24 36	64 4 34 52	104 D 44 68	124 T 54 84	144 d 64 100	164 t 74 116
0 1 0 1	5 ENQ 5 5	25 NAK 15 21	45 % 25 37	65 5 35 53	105 E 45 69	125 U 55 85	145 e 65 101	165 u 75 117
0 1 1 0	6 ACK 6 6	26 SYN 16 22	46 & 26 38	66 6 36 54	106 F 46 70	126 V 56 86	146 f 66 102	166 v 76 118
0 1 1 1	7 BEL 7 7	27 ETB 17 23	47 ' 27 39	67 7 37 55	107 G 47 71	127 W 57 87	147 g 67 103	167 w 77 119
1 0 0 0	10 BS 8 8	30 CAN 18 24	50 (28 40	70 8 38 56	110 H 48 72	130 X 58 88	150 h 68 104	170 x 78 120
1 0 0 1	11 HT 9 9	31 EM 19 25	51) 29 41	71 9 39 57	111 I 49 73	131 Y 59 89	151 i 69 105	171 y 79 121
1 0 1 0	12 LF A 10	32 SUB 1A 26	52 * 2A 42	72 : 3A 58	112 J 4A 74	132 Z 5A 90	152 j 6A 106	172 z 7A 122
1 0 1 1	13 VT B 11	33 ESC 1B 27	53 + 2B 43	73 ; 3B 59	113 K 4B 75	133 [5B 91	153 k 6B 107	173 { 7B 123
1 1 0 0	14 FF C 12	34 FS 1C 28	54 , 2C 44	74 < 3C 60	114 L 4C 76	134 \ 5C 92	154 l 6C 108	174 ¦ 7C 124
1 1 0 1	15 CR D 13	35 GS 1D 29	55 – 2D 45	75 = 3D 61	115 M 4D 77	135] 5D 93	155 m 6D 109	175 } 7D 125
1 1 1 0	16 SO E 14	36 RS 1E 30	56 . 2E 46	76 > 3E 62	116 N 4E 78	136 ^ 5E 94	156 n 6E 110	176 ~ 7E 126
1 1 1 1	17 SI F 15	37 US 1F 31	57 / 2F 47	77 ? 3F 63	117 O 4F 79	137 – 5F 95	157 o 6F 111	177 RUBOUT (DEL) 7F 127

KEY

octal → 5
ENQ ← ASCII character
hex → 5   5 ← decimal

Tektronix

REF: ANSI STD X3.4-1977
IEEE STD 488.1-1987
ISO STD 646-2973

Appendix C: Algorithms

The waveform analyzer can take many automatic measurements and perform a variety of other calculations. By knowing how they make these calculations, you may better understand how to use your waveform analyzer and how to interpret the results.

Measurement Variables

The waveform analyzer uses a variety of variables in its calculations. This section discusses each variable and how to set it.

High and Low

High is the value used as the 100% level in measurements such as fall time and rise time. For example, if you request the 10% to 90% rise time, then the waveform analyzer calculates 10% and 90% as percentages with High representing 100%.

Low is the value used as the 0% level in measurements such as fall time and rise time.

The exact meaning of High and Low depends on which calculation method you choose. To set the method used to determine High you use the CALCulate:WMParameter:HMEThod command. To set the method used to determine Low you use the CALCulate:WMParameter:LMEThod command. The methods are PEAK, MODE, AUTO, and ABSolute.

PEAK defines the 0% and the 100% waveform levels as the lowest amplitude (most negative) and the highest amplitude (most positive) samples. The PEAK method is useful for measuring frequency, width, and period for many types of signals. PEAK is sensitive to waveform ringing and spikes, however, and does not always accurately measure rise time, fall time, overshoot, and undershoot.

MODE attempts to find the highest density of points above and below the waveform midpoint. It attempts to ignore ringing and spikes when determining the 0% and 100% levels. This method works well when measuring square waves and pulse waveforms.

The waveform analyzer calculates the histogram-based High and Low values as follows:

1. It makes a histogram of the record with 256 bins.

2. It splits the histogram into two sections at the halfway point between Min and Max (also called Mid).
3. The level with the most points in the upper histogram is the High value, and the level with the most points in the lower histogram is the Low value. (Choose the levels where the histograms peak for High and Low.)

   If Mid gives the largest peak value within the upper or lower histogram, then return the Mid value for both High and Low (this is probably a very low amplitude waveform).

   If more than one histogram level (bin) has the maximum value, choose the bin farthest from Mid.

This algorithm does not work well for two-level waveforms with greater than about 100% overshoot.

AUTO attempts to use the MODE method but will switch to the PEAK method if the histogram does not show an obvious consistent high level. For example, the MODE histogram operating on a triangle wave would not find significant High and Low levels so AUTO would select PEAK. On a square wave AUTO would pick the MODE method.

ABSolute uses the absolute value set with the commands CALCulate:WMParameter:HIGH or CALCulate:WMParameter:LOW.

Measurement Reference Levels

You can set the various reference levels used to take the automated measurements. You can choose to set reference levels in absolute vertical units or in relative units of percent or ratio. Use the command CALCulate:WMParameter:RMEThod to choose the reference method. The reference levels are as follows:

HREFerence the waveform high reference or distal level. Used in fall time and rise time calculations. In the RELative mode, you use the command CALCulate:WMParameter:HREFerence:RELative to set it from 0% to 100%, with a reset value of 90%. You can set it to a voltage level with the command CALCulate:WMParameter:HREFerence.

MREFerence the waveform middle reference or mesial level. In the RELative mode, you use the command CALCulate:WMParameter:MREFerence:RELative to set MREFerence from 0% to 100%, with a reset value of 50%. You can set it to a voltage level with the command CALCulate:WMParameter:MREFerence. You can also specify a hysteresis value for the MREFerence that reduces the effects of noise on measurements. The HYSTeresis value is a percent or ratio of the AMPLitude value.

LREFerence the waveform low reference or proximal level. Used in fall time and rise time calculations. In the RELative mode, you use the command CALCulate:WMParameter:LREFerence:RELative to set it from 0% to 100%, with a reset value of 10%. You can set it to a voltage level with the command CALCulate:WMParameter:LREFerence.

Other Variables

The waveform analyzer also measures several values itself that it uses to help calculate measurements.

RecordLength is the number of data points in the waveform record.

Hysteresis reduces the effects of noise on measurements by providing a guard band of 5% (the default) of the waveform amplitude, above and below the midpoint value. Hysteresis can be set in the range 0% to 50%. It is used in MCross1, MCross2, and MCross3 calculations.

For example, once a crossing has been measured in a negative direction, the waveform data must fall below 5% of the amplitude from the MidRef point before the measurement system is armed and ready for a positive crossing. Similarly, after a positive MidRef crossing, waveform data must exceed 5% of the amplitude before a negative crossing can be measured. Hysteresis is useful when you are measuring noisy signals, because it allows the waveform analyzer to ignore minor fluctuations in the signal.

MCross Calculations

MCross1, MCross2, and MCross3 refer to the first, second, and third MidRef cross times, respectively. (See Figure C–1.)

The polarity of the crossings does not matter for these variables, but the crossings alternate in polarity; that is, MCross1 could be a positive or negative crossing, but if MCross1 is a positive crossing, MCross2 will be a negative crossing.

The waveform analyzer calculates these values as follows:

1. Find the first MidRefCrossing in the waveform record (or the gated region). This is MCross1.
2. Continuing from MCross1, find the next MidRefCrossing in the waveform record (or the gated region) of the opposite polarity of MCross1. This is MCross2.
3. Continuing from MCross2, find the next MidRefCrossing in the waveform record (or the gated region) of the same polarity as MCross1. This is MCross3.

MCross1Polarity is the polarity of first crossing (no default). It can be rising or falling.

StartCycle is the starting time for cycle measurements. It is a floating-point number with values between 0.0 and (RecordLength – 1.0), inclusive.

StartCycle = MCross1

EndCycle is the ending time for cycle measurements. It is a floating-point number with values between 0.0 and (RecordLength – 1.0), inclusive.

EndCycle = MCross3

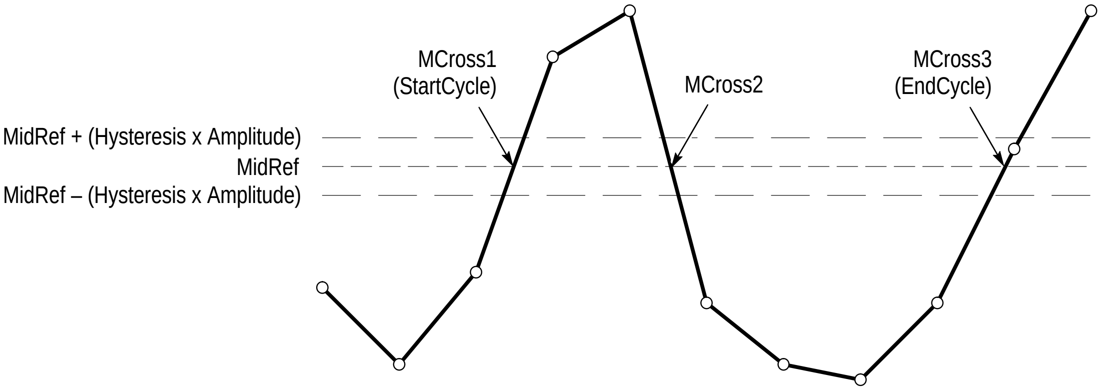

MCross1
(StartCycle)
MCross2
MCross3
(EndCycle)
MidRef + (Hysteresis x Amplitude)
MidRef
MidRef – (Hysteresis x Amplitude)

Figure C–1: MCross Calculations

Waveform[<0.0 ... RecordLength–1.0>] — holds the acquired data.

Measurement Zone and Edge Selection

The waveform analyzer can take measurements over the entire waveform or over a user-specified measurement zone. Some measurements also require that the user specify the waveform edge to be measured. Usage of the commands needed to specify zoned (also called gated) measurements is discussed in the section *Measurements* in chapter 3. This section briefly describes each zoned measurements and waveform edge selection.

Zoned Measurements

The `CALCulate<n>:WMParameter:GATE` commands control whether zoned measurements are on or off:

- When `CALC<n>:WMP:GATE` is set to `ON`, measurements are taken within a measurement zone defined using other parameters to the `:GATE` commands.

- When CALC<n>:WMP:GATE is set to OFF, the measurement zone becomes the entire waveform record. The measurement zone defined by the :GATE command parameters is ignored.

All measurements included in the CALC block are affected by these commands, but gating for each CALC block is settable independently.

Edge Selection

The CALCulate<n>:WMParameter:EDGE command selects the waveform edge that the measurement is taken on for these edge-based measurements:

- CROSs, NCROSs, and PCROSs
- DELay
- FTIMe and RTIMe
- COPulse

The CALCulate<n>:WMParameter:SLOPE command selects the polarity of the edges used in DELay measurements; the polarity of the edge used for the other edge measurements is determined implicitly (for example, RTIME use positive slope, etc.).

When CALC:WFM:GATE is set to OFF, the waveform analyzer finds the edge within the entire waveform record; when set to ON, the waveform analyzer finds the edge with in the measurement zone.

Measurement Algorithms

The automated measurements are defined and calculated as follows.

Amplitude

Amplitude = High – Low

Area

The arithmetic area of one waveform or of its measurement zone. Remember that one waveform is not necessarily equal to one cycle. For cyclical data you may prefer to use the cycle area rather than the arithmetic area.

if Start = End then return the (interpolated) value at Start.

Otherwise,

$$Area = \int_{Start}^{End} Waveform(t)\mathrm{dt}$$

For details of the integration algorithm, see page C–17.

COPulse

Center-Of-Pulse timing-measurement. Returns the average of the times, relative to the trigger point, of the three measurement reference levels on the pulse leading edge that contains MCRoss1 and the three levels on the pulse trailing edge containing MCRoss2. See *Mcross Calculations* on page C–3 for definitions of crossings. See *Measurement Reference Levels* on page C–2 for descriptions of the reference levels.

COPulse used the edge index set by `CALCulate<n>:WMParameter:EDGE` as the leading edge and next edge of opposite polarity as the trailing edge.

Cross

Timing measurement. The time relative to the trigger point at which the crossing that you specify occurs. The CROSs measurement searches for the Nth occurrences of an edge; during the search it counts edges of either polarity.

1. Searching from Start to End of waveform record, find the first transition — either negative-going or positive-going — through MREF (middle ref).

2. Continue the search process until the Nth crossing is found (user specifies N using the `CALCulate:WMParameter:EDGE` command).

3. Cross = Time@crossing, where Time@trigger = 0.

Pretrigger crossings return negative times; posttrigger crossings return positive times.

Positive values for N force the search at the start of the waveform record; Negative values and zero at the end (zero designates the last crossing, –1 the next to the last crossing, and so on in the waveform record).

Cycle Area

Amplitude (voltage) measurement. The area over one waveform cycle. For non-cyclical data, you might prefer to use the Area measurement.

If StartCycle = EndCycle then return the (interpolated) value at StartCycle.

$$CycleArea = \int_{StartCycle}^{EndCycle} Waveform(t)\mathrm{dt}$$

For details of the integration algorithm, see page C–17.

Cycle Mean

Amplitude (voltage) measurement. The mean over one waveform cycle. For non-cyclical data, you might prefer to use the Mean measurement.

If StartCycle = EndCycle then return the (interpolated) value at StartCycle.

$$CycleMean = \frac{\int_{StartCycle}^{EndCycle} Waveform(t)\mathrm{dt}}{(EndCycle - StartCycle) \times SampleInterval}$$

For details of the integration algorithm, see page C–17.

Cycle RMS

The true Root Mean Square voltage over one cycle.

If StartCycle = EndCycle then CycleRMS = Waveform[Start].

Otherwise,

$$CycleRMS = \sqrt{\frac{\int_{StartCycle}^{EndCycle} (Waveform(t))^2\mathrm{dt}}{(EndCycle - StartCycle) \times SampleInterval}}$$

For details of the integration algorithm, see page C–17.

Delay

Timing measurement. The amount of time between the *MidRef* crossings of two different traces or two different places on the same trace.

Delay measurements are actually a group of measurements. To get a specific delay measurement, you must specify the target and reference crossing polarities and edges.

Gain

Ratio of two amplitudes measurement. An amplitude measurement (see *Amplitude* on page C–5) is taken of the reference and target waveforms.

$$Gain = \frac{Amplitude_{target}}{Amplitude_{refernece}}$$

Fall Time

Timing measurement. The time taken for the falling edge of a pulse to drop from a HighRef value (default = 90%) to a LowRef value (default = 10%).

Figure C–2 shows a falling edge with the two crossings necessary to calculate a fall time measurement.

1. Based on the index n specified by the `CALCulate:WMParameter:EDGE` command, count edges according to the following rules:

   - Positive arguments search forward from the start-to-end of the waveform record; zero and negative arguments search backward from the end-to-start.

   - Zero specifies the last edge in the waveform record.

   - Count only edges that go through both HREF and LREF (see Figure C–2).

2. Continue the search process until the nth negative-going edge is found (user specifies n using the `CALCulate:WMParameter:EDGE` command).

3. Find the time of the HREF crossing for the nth edge. This time is THF. (Use linear interpolation if necessary.)

4. Find the time of the LREF crossing for the nth edge. This time is TLF. (Use linear interpolation if necessary.)

5. FallTime = TLF – THF

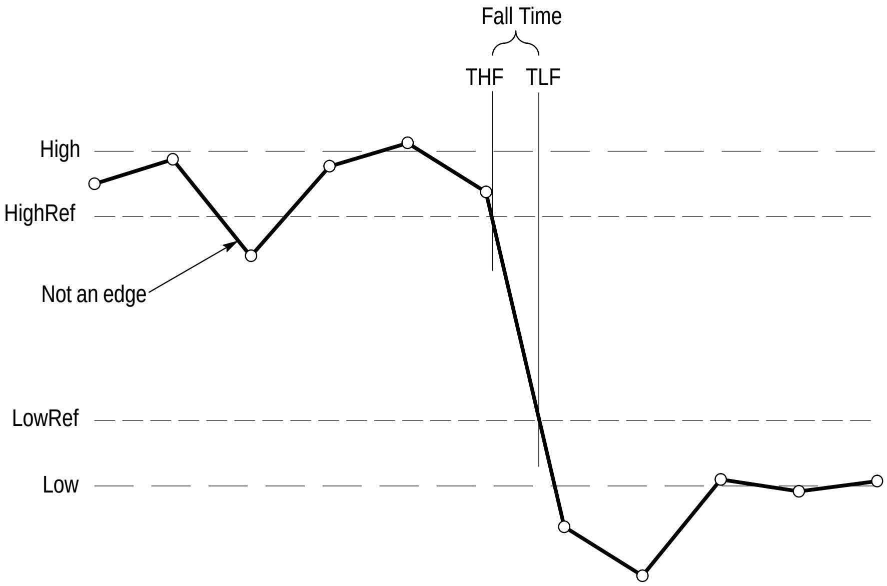


Figure C–2: Fall time

Frequency

Timing measurement. The reciprocal of the period. Measured in Hertz (Hz) where 1 Hz = 1 cycle per second.

If Period = 0 or is otherwise bad, return an error.

Frequency = 1/Period

High

100% (highest) voltage reference value. (See *High and Low* on page C–1.)

Using the min-max measurement technique:

High = Max

Low

0% (lowest) voltage reference value calculated. (See *High and Low* on page C–1.)

Using the min-max measurement technique:

Low = Min

Maximum

Amplitude (voltage) measurement. The maximum voltage. Typically the most positive peak voltage.

Examine all Waveform[ ] samples from Start to End inclusive, and set Max equal to the greatest magnitude Waveform[ ] value found.

Mcross1
Mcross2
Mcross3

Timing-measurement functions. These three measurements are of the times relative to the trigger point at which the first, second, and third Mid Ref crossings occur in the waveform. See *Mcross Calculations* on page C–3 for definitions of crossings.

Mcross1, Mcross2, and Mcross3 cannot be included as measurements in the measurement lists AAMList or WMList. Rather they can be used as functions in calculation expressions.

Pretrigger crossings return negative times; posttrigger crossings return positive times.

Mean

The arithmetic mean for one waveform. Remember that one waveform is not necessarily equal to one cycle. For cyclical data you may prefer to use the cycle mean rather than the arithmetic mean.

$$Mean\ =\sum_{i=0}^{i=RecordLength-1}\frac{Waveform\ [i]}{RecordLength}$$

Minimum

Amplitude (voltage) measurement. The minimum amplitude. Typically the most negative peak voltage.

Examine all Waveform[ ] samples from Start to End inclusive, and set Min equal to the smallest magnitude Waveform[ ] value found.

Negative Duty Cycle

Timing measurement. The ratio of the negative pulse width to the signal period expressed as a percentage.

NegativeWidth is defined in **Negative Width**, below.

If Period = 0 or undefined then return an error.

$$NegativeDutyCycle = \frac{NegativeWidth}{Period} \times 100\%$$

Negative Width

Timing measurement. The distance (time) between MidRef (default = 50%) amplitude points of a negative pulse.

```
    If MCross1Polarity = '–'
then
    NegativeWidth = (MCross2 – MCross1)
else
    NegativeWidth = (MCross3 – MCross2)
```

Ncross

Timing measurement. The time relative to the trigger point at which the negative-going crossing that you specify occurs. The NCROSs measurement searches for the Nth occurrence of an edge; during the search it counts only negative edges.

1. Searching from Start to End of waveform record, find the first negative-going transition through MREF (middle reference).

2. Continue the search process until the Nth negative-going crossing is found (user specifies N using the `CALCulate:WMParameter:EDGE` command).

3. NCross = Time@ncrossing, whereTime@trigger = 0.

   Pretrigger crossings return negative times; posttrigger crossings return positive times.

Positive values for N force the search at the start of the waveform record; Negative values and zero, at the end (zero designates the last crossing, –1 the next to the last crossing, and so on in the waveform record).

Overshoot

Amplitude (voltage) measurement. Overshoot finds the first positive-going edge in the waveform record (or gated area) and calculates value as follows:

$$Overshoot = \frac{Max - High}{Amplitude} \times 100\%$$

This measurement ignores settings of the `CALCulate:WMParameter:EDGE` and :SLOPE commands when locating the edge to measure.

Pcross

Timing measurement. The time relative to the trigger point at which the positive-going crossing that you specify occurs. The PCROSs measurement searches for the N^{th} occurrence of an edge; during the search it counts only positive edges.

1. Searching from Start to End of waveform record, find the first positive-going transition through MREF (middle reference).
2. Continue the search process until the N^{th} positive-going crossing is found (user specifies N).
3. PCross = Time@pcrossing, whereTime@trigger = 0.

   Pretrigger crossings return negative times; posttrigger crossings return positive times.

Positive values for N force the search at the start of the waveform record; Negative values and zero, at the end (zero designates the last crossing, –1 the next to the last crossing, and so on in the waveform record).

Peak to Peak

Amplitude measurement. The absolute difference between the maximum and minimum amplitude.

PeaktoPeak = Max – Min

Period

Timing measurement. Time taken for one complete signal cycle. The reciprocal of frequency. Measured in seconds.

Period = MCross3 – MCross1

Phase

Timing measurement. The amount of phase shift, expressed in degrees of the target waveform cycle, between the *MidRef* crossings of two different waveforms. Waveforms measured should be of the same frequency or one waveform should be a harmonic of the other.

Phase is a dual waveform measurement; that is, it is measured from a target waveform to a reference waveform. To get a specific phase measurement, you must specify the target and reference sources.

Phase is determined in the following manner:

1. The first *MidRefCrossing (MCross1Target)* and third *(MCross3*) in the source (target) waveform are found.
2. The period of the target waveform is calculated (see *Period* above).
3. The first *MidRefCrossing (MCross1Ref)* in the reference waveform crossing in the same direction (polarity) as that found *MCross1Target* for the target waveform is found.
4. The phase is determined by the following:

$$Phase = \frac{MCross1Ref - MCross1Target}{Period} \times 360$$

If the target waveform leads the reference waveform, phase is positive; if it lags, negative.

Positive Area

Amplitude (voltage) measurement. The arithmetic area over the absolute value of one waveform or if measurement gating is on, over its gated area. Remember that one waveform is not necessarily equal to one cycle. For cyclical data you may prefer to use the cycle area rather than the arithmetic area.

if Start = End then return the (interpolated) value at Start.

Otherwise,

$$Area = \int_{Start}^{End} \mathrm{ABS}(Waveform(t))\mathrm{dt}$$

For details of the integration algorithm, see page C–17.

Positive Cycle Area

Amplitude (voltage) measurement. The area over the absolute value of one waveform cycle. For non-cyclical data, you might prefer to use the Area measurement.

If StartCycle = EndCycle then return the (interpolated) value at StartCycle.

$$CycleArea = \int_{StartCycle}^{EndCycle} \mathrm{ABS}(Waveform(t))\mathrm{dt}$$

For details of the integration algorithm, see page C–17.

Positive Duty Cycle

Timing measurement. The ratio of the positive pulse width to the signal period, expressed as a percentage.

PositiveWidth is defined in **Positive Width**, following.

If Period = 0 or undefined then return an error.

$$PositiveDutyCycle = \frac{PositiveWidth}{Period} \times 100\%$$

Positive Width

Timing measurement. The distance (time) between MidRef (default = 50%) amplitude points of a positive pulse.

If MCross1Polarity = ‘+’

then

PositiveWidth = (MCross2 – MCross1)

else

PositiveWidth = (MCross3 – MCross2)

Preshoot

Amplitude (voltage) measurement. Preshoot finds the first positive-going edge in the waveform record (or gated area) and calculates value as follows:

$$Preshoot = \frac{Low\ -\ Min}{Amplitude} \times 100\%$$

This measurement ignores settings of the `CALCulate:WMParameter:EDGE` and :SLOPE commands when locating the edge to measure.

Rise Time

Timing measurement. Time taken for the leading edge of a pulse to rise from a LowRef value (default = 10%) to a HighRef value (default = 90%).

Figure C–3 shows a rising edge with the two crossings necessary to calculate a rise time measurement.

1. Based on the index n specified by the `CALCulate:WMParameter:EDGE` command, count edges according to the following rules:

   - Positive arguments search forward from the start-to-end of the waveform record; zero and negative arguments search backward from the end-to-start.
   - Zero specifies the last edge in the waveform record.
   - Count only edges that go through both HREF and LREF (see Figure C–2).

2. Continue the search process until the n^{th} positive-going edge is found_(user specifies n using the `CALCulate:WMParameter:EDGE` command).
3. Find the time of the HREF crossing for the n^{th} edge. This time is THF. (Use linear interpolation if necessary.)
4. Find the time of the LREF crossing for the n^{th} edge. This time is TLF. (Use linear interpolation if necessary.)
5. RiseTime = THR – TLR

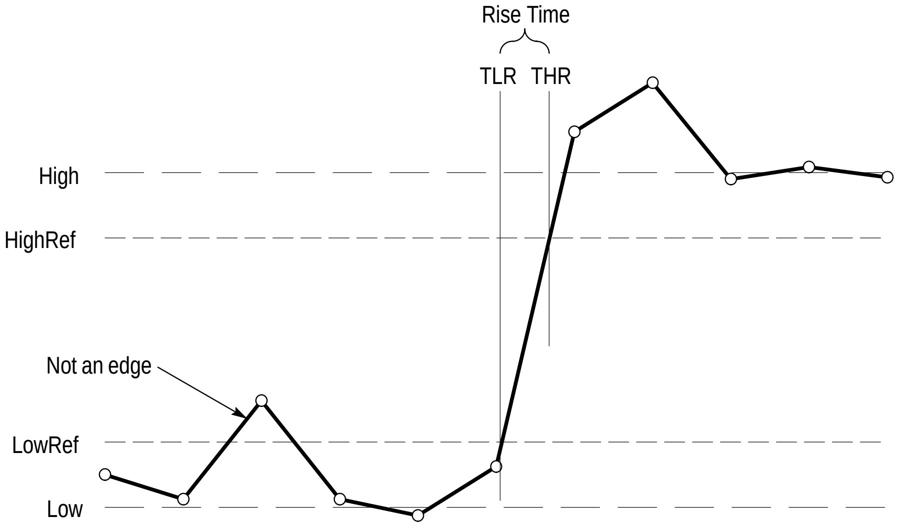


Figure C–3: Rise Time

RMS:

Amplitude (voltage) measurement. The true Root Mean Square voltage.

If Start = End then RMS = the (interpolated) value at Waveform[Start].

Otherwise,

$$RMS = \sqrt{\frac{\int_{Start}^{End} (Waveform(t))^2 \mathrm{dt}}{(End - Start) \times SampleInterval}}$$

For details of the integration algorithm, see *Integration Algorithm* in this section.

TTrig Timing measurement. The time difference between the main and the delay triggers.

TTrig = Time@delaytrig – Time@maintrigger

Value returned is independent of channel number. Value returned is valid only when the delay trigger source is not set to immediate.

Differentiation Algorithm

The differentiation algorithm used by the waveform analyzer is as follows:

$$Diff(w(n)) = 0$$

$$\quad for\ n = 0$$

$$Diff(w(n)) = [w(n + 1) - w(n - 1)]/(2T)$$

$$\quad for\ 1 \le n \le (R-1)$$

$$Diff(w(n)) = [w(R-1) - w(R-2)]/T$$

$$\quad for\ n = (R-1)$$

where:

n = index into the record of data points
$w(n)$ = input sampled data point
T = time interval between successive samples
R = record length

Integration Algorithm

The integration algorithm used by the waveform analyzer is as follows:

$$Intg(w(n)) = 0$$

$$\quad for\ n = 0$$

$$Intg(w(n)) = \left[1/2\ w(0) + \sum_{m=1}^{n-1} w(m) + 1/2\ w(n) \right] \times T$$

$$\quad for\ 1 \le n \le R$$

where:

n = index into record of data points
$w(n)$ = input sampled data point
T = time interval between successive samples
R = record length in points

Smooth Algorithm

The smoothing algorithm used by the waveform analyzer is as follows:

$$Smooth(w(n)) = (1/s)\left[\sum_{m=0}^{n+h} w(m) + (h - n) \times w(0)\right]$$

$$for\ n < h$$

$$Smooth(w(n)) = (1/s)\left[\sum_{m=n-h}^{n+h} w(m)\right]$$

$$for\, h \le n \le R-1-h$$

$$Smooth(w(n)) = (1/s)\left[\sum_{m=n-h}^{R-1} w(m) + (R-1-n) \times w(R-1)\right]$$

$$for\ n > R-1-h$$

where:

n = index into record of data points
$w(n)$ = input sampled data point
s = smoothing interval in samples; the second argument
h = half interval: (s – 1)/2 rounded down
R = record length in points

The smoothed waveform is derived by computing the average value of the corresponding point of the original waveform and a certain number of points of the original waveform on either side of the corresponding point. The number of points on either side is derived from the smoothing interval, which you set with the command CALC:SMO:POIN.

Near the ends of the waveform, nonexistent points beyond the ends of the waveform are required for averaging. The nonexistent points are assumed to be the value of the corresponding end points. This method of extending the waveform is arbitrary, so the results within a smoothing interval of the ends of the waveform must be interpreted accordingly.

Digital Filter Algorithms

This section describes how the digital filter of the waveform analyzer operates. The commands in the CALCulate:FILTer subsystem control the digital filter.

An Ideal Filter

The filter functions in the waveform analyzer instrument allow lowpass, highpass, bandpass and notch filters to be applied to any acquired set of data. A perfect filter would have unity transmission (with linear phase response) in the pass band, infinite attenuation in the stop band and abruptly change from pass to stop band. The transfer function for an ideal bandpass filter is depicted in Figure C–4.

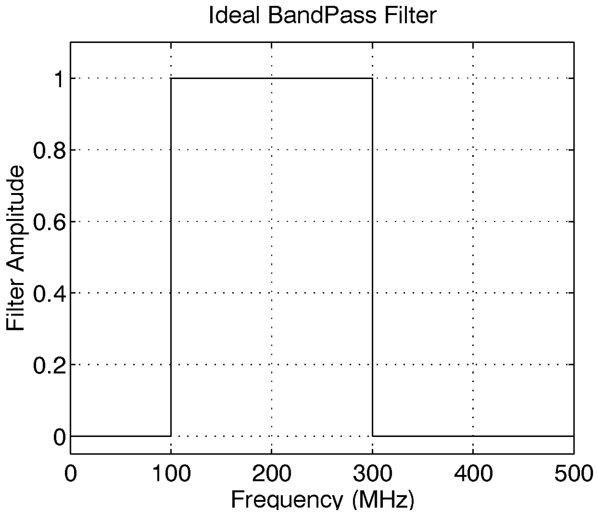


Figure C–4: Transfer function H(f) for an ideal bandpass filter

When you use a filter in the waveform analyzer, the frequency response of the desired filter is inverse Fourier transformed to calculate the response of the filter for a time domain impulse and this impulse response is convolved with the waveform data as shown in the following equation:

$$h(t) = \mathcal{F}^{-1}\{H(f)\}$$

$$output\ \ wfm = (input\ \ wfm) * h(t)$$

These equations are mathematically correct, however, it is impossible to implement them. For any ideal filter, which has abrupt changes in the transfer function, the impulse response extends for all time. Clearly an infinitely long impulse response cannot be convolved with the waveform data.

Rectangular Window

One possible solution for dealing with the infinitely long impulse function, h(t) is to reduce it to a manageable length. The simplest technique is to use the central points of the filter and throw away the remaining points. What this does is apply a time domain rectangular window filter to the impulse response h(t). Figure C–5 shows the transfer function for an ideal lowpass filter. Figure C–6 shows the time domain impulse for the lowpass filter and depicts one possible rectangular window which selects the central filter points.

Truncating the infinitely long impulse response of the filter to a finite length results in a filter frequency response that is no longer ideal. In the time domain the filter impulse response has been multiplied by a window w(t):

$$new\ h(t) = h(t) \cdot w(t)$$

To see how the frequency domain transfer function for the filter, H(f), has changed it is necessary to transfer the above equation to the frequency domain. Multiplication in the time domain corresponds to convolution in the frequency domain.

$$W(f) = \mathcal{T}\{w(t)\}$$

$$New\ H(f) = H(f) * W(f)$$

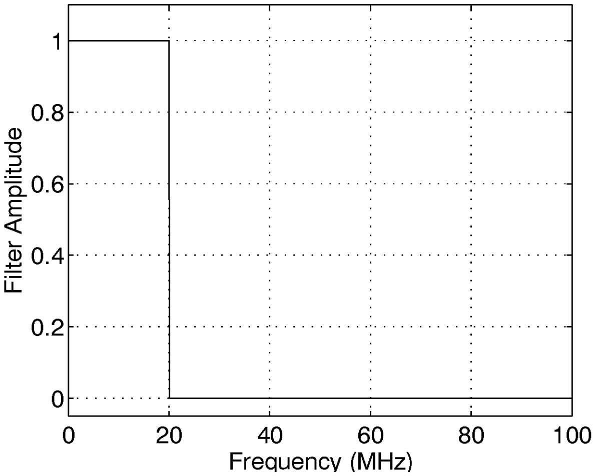


Figure C–5: Transfer function for an ideal lowpass filter

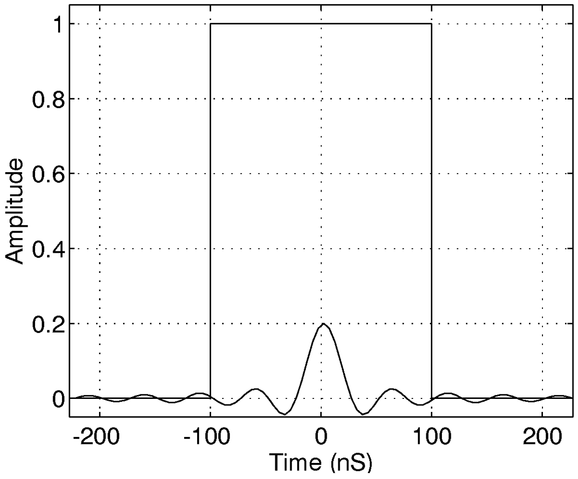


Figure C–6: Using a rectangular window to truncate the data from Figure C–5 to a finite number of points

The frequency domain convolution of H(f) with W(f) has three effects: the filter edges are no longer abrupt, the pass band transmission is no longer exactly unity, and the stop band attenuation is no longer infinite. Figures C–7 through C–9 show the original, ideal filter and the resultant filter with three different lengths of rectangular windows. Note that these plots use a dB scale for the filter.

As more points are used in the filter (corresponding to a longer window) the transition becomes sharper and sharper. However, in this example, the worst case attenuation of the filter in the stop band stays fixed at about –21 dB. Examining Figures C–7 through C–9 carefully, you can see that the peak amplitude of the ripple in the stop band (i.e., minimum attenuation) remains fixed at about –21 dB. As more points are used, the filter becomes sharper but this side lobe remains 21 dB down. For other filter shapes, such as bandpass and notch filters, the worst case stop band attenuation can be as low as –15 dB. The limitation on the stop band attenuation is the major drawback of using a rectangular window.

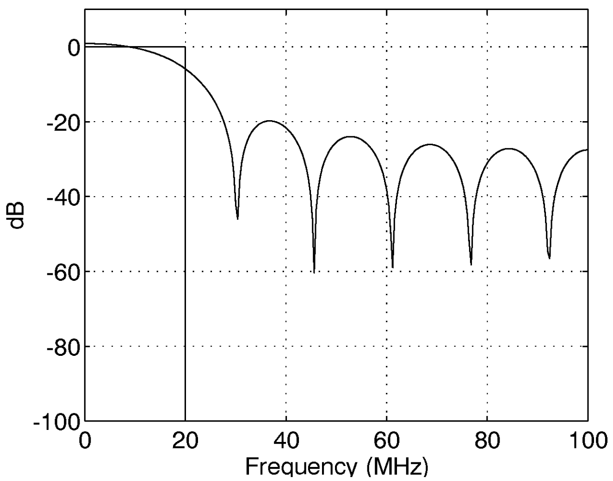


Figure C–7: Lowpass filter transfer function obtained by truncating the impulse response to just a few points

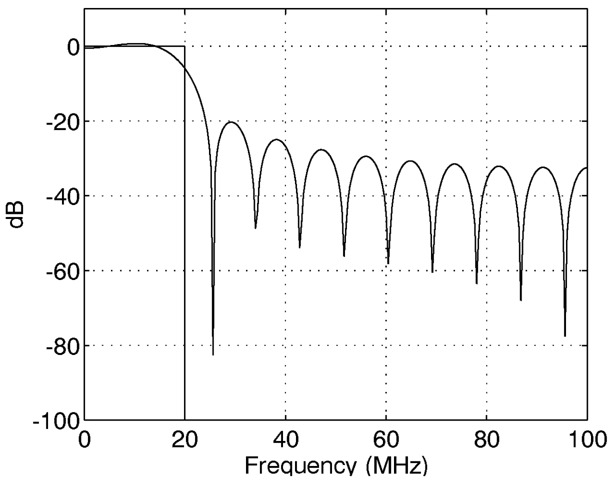


Figure C–8: Using more points in the Lowpass filter results in a steeper transition at the cutoff frequency

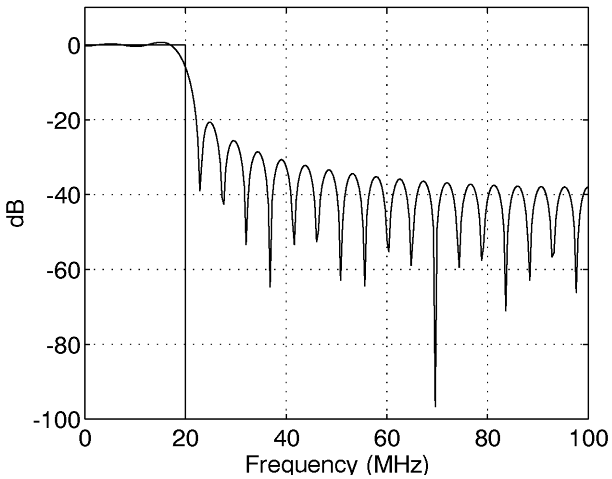


Figure C–9: Using many more points in the Lowpass filter results in a quicker transition but a minimum attenuation of 21 dB

Kaiser Window

When the filter response is truncated with a rectangular window the minimum attenuation in the stop band is at best 21 dB. In order to achieve greater attenuation in the stop band a non–rectangular window must be applied to the filter data.

There are many choices for non–rectangular windows. Common windows include Bartlett, Hamming, Hanning, and Blackman. The filter in the waveform analyzer employs a Kaiser window. This window was chosen because it offers a range of possible window shapes, and thus different stop band attenuations. For a window that is M+1 points long, the Kaiser window is defined as follows:

$$w[n] = \left\{ \begin{array}{ll} \dfrac{I_0 \left[\beta \left(1 - \left[\frac{(n-M/2)}{M/2} \right]^2 \right)^{1/2} \right]}{I_0 \left[\beta \right]} & 0 \le n \le M \\ 0 & otherwise \end{array} \right\}$$

I_0 represents the zero order modified Bessel function of the first kind. β is a parameter that ranges from 0 to infinity. The larger the value of β, the more the window tapers at the edges. When β=0 the Kaiser window reduces to a rectangular window. Figure C–10 shows three Kaiser windows with 200 points in the window and a β of 1, 5 and 20.

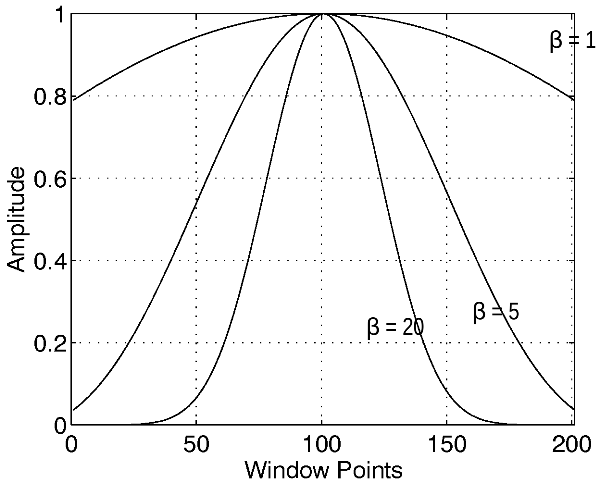


Figure C–10: Kaiser window with 200 points and β = 1, 5 and 20

For larger values of β, the Kaiser window tapers off slowly towards the edges of the window. Using the same number of data points and taking a Fourier transform of a Kaiser window and a rectangular window, the transform of the Kaiser window is broader than the rectangular window but the side lobes are much farther down.

As stated in the last section, the frequency domain transfer function of the filter is given by convolving the transfer function of the ideal filter with that of the window.

$$W(f) = \mathcal{T}\{w(t)\}$$

$$New\ H(f) = H(f) * W(f)$$

If a Kaiser window is used with the same number of points as a rectangular window, then the transition width will not be as narrow but the minimum stop band attenuation will be much greater than the 21 dB achieved with a rectangular window. To graphically see this effect, refer to Figure C–11 which shows a lowpass filter obtained with a Kaiser window with β=2.65. Compare the transfer function of this filter with that in Figure C–9 where the same number of points were used but with a rectangular window.

By setting β high enough, the stop band attenuation can be increased. The cost for this increase is a wider filter transition region which can be countered by using more points in the filter. As pointed out previously, the greater the number of points in the filter, the narrower the filter transition region. However, there is a limit to the number of points in the filter. As described more fully in the section on edge effects, the number of points in the filter is limited to a maximum of 10% of the record length.

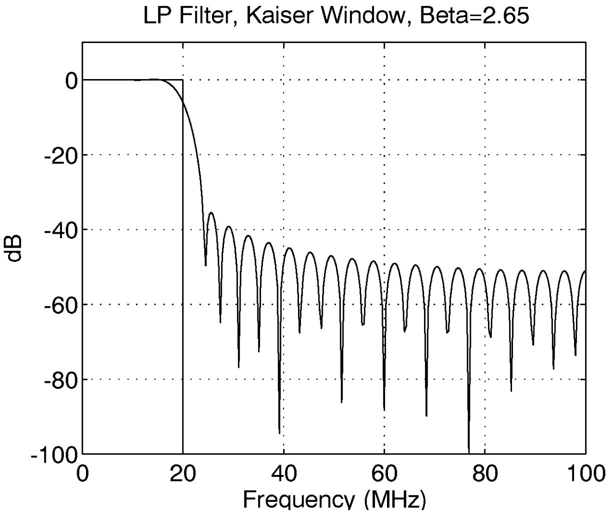


Figure C–11: Compare this result with Figure C–9 with the same number of points but a rectangular window

Defining Filter Specifications

The waveform analyzer filter is specified in a manner which may be unfamiliar to those used to working with analog filters. One difference is that cutoff and start/stop frequencies are specified as the –6 dB point, not the –3 dB point. Figure C–12 shows waveform analyzer specifications for a lowpass filter.

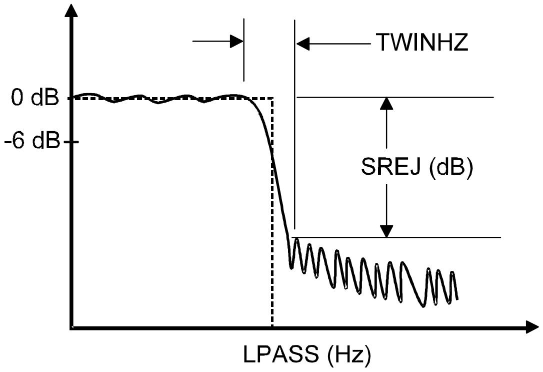


Figure C–12: Filter specifications for a lowpass filter

The cutoff frequency of the filter is specified in Hertz with the LPAS command. As an example, to set a lowpass cutoff frequency of 20 MHz, use the command:

```
CALC1:FILT:FREQ:LPAS 20E6
```

LPAS specifies a lowpass filter and 20E6 sets the cutoff frequency (at –6 dB).

The stop band attenuation or stop band rejection is set with the SREJ command. The SREJ is given in dB. The minimum attenuation is 15 dB and the maximum is 100 dB (the default value is 60 dB). As an example, to set the stop band attenuation to 40 dB use the command:

```
CALC1:FILT:FREQ:SREJ 40
```

The final specification of the filter is the relative filter transition width, TWID. The TWID is directly related to the TWIDHZ depicted in Figure C–12. TWIDHZ is a measure of how quickly the filter response changes from pass band to stop band. The TWID is specified relative to the Nyquist frequency for the acquisition record. The Nyquist frequency is defined as:

$$F_{NYQ} = \frac{1}{2 \cdot TINT}$$

Where TINT is the sample interval. Equivalently, the Nyquist frequency is also equal to 1/2 of the sampling rate.

The TWID is then defined as:

$$TWID = \frac{TWIDHZ}{F_{NYQ}} = TWIDHZ \cdot 2 \cdot TINT$$

Valid ranges for TWID are between 0 and 1. The smaller the value, the narrower the transition region. The default value is 0.1. As an example, to set the TWID to 0.05 use the command:

```
CALC1:FILT:FREQ:TWID 0.05
```

Define highpass filter in the same manner as the lowpass filter. The only difference is that it is necessary to specify a HPAS frequency. As an example, to set the highpass frequency to 100 MHz, use the command:

```
CALC1:FILT:FREQ:HPAS 100E6
```

Figure C–13 depicts the specifications for a band pass filter. The same specifications are used for the notch filter, but the two filters have swapped pass band and stop band regions. Like the lowpass filter, a bandpass filter has a specification for SREJ and TWID. SREJ and TWID are equal on both sides of the bandpass region. For the bandpass and notch filters it is necessary to specify a start and stop frequency. This is done with the STAR and STOP commands. As an example to set up a bandpass filter with a start frequency of 50 MHz and a stop frequency of 75 MHz use the command:

```
CALC1:FILT:FREQ:BPAS; STAR 50E6; STOP 75E6
```

Instead of specifying a STAR and STOP frequency, it is possible to specify CENT and SPAN frequencies.

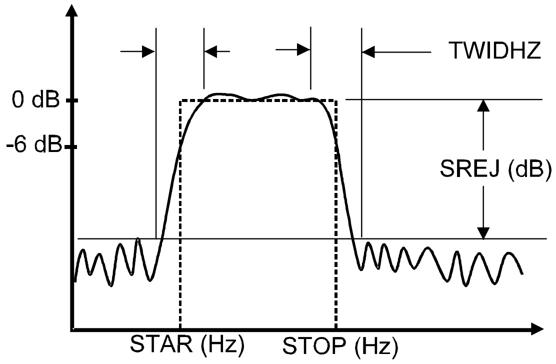


Figure C–13: Filter specifications for a bandpass filter

Filter Length Limitations and Edge Effects

After the TWID and the SREJ are specified, the β for the window is calculated from the following expressions:

$$SATT = SREJ + 6.0206$$

$$\beta = \begin{cases} 0.1102(SATT-8.7) & SATT > 50 \\ 0.58422(SATT-21)^{0.4} + 0.07886(SATT-21) & 21 \le SATT \le 50 \\ 0 & SATT < 21 \end{cases}$$

The value 6.0206 is added to SREJ to ensure that the additive ripple from the notch and bandpass filters stay within specifications. Adding 6.0206 to SREJ is equivalent to dividing the ripple specification by two.

The number of points in the filter are calculated from the expression:

$$NFILT = \frac{SATT-8}{2.285 \cdot \pi \cdot TWID} + 1$$

Actually the waveform analyzer algorithm uses the smallest odd integer greater than the NFILT calculated in the above expression (if NFILT=75.1 then 77 would be used). If there is a high level of stop band attenuation, SREJ, or if the relative filter transition width, TWID, is set very small then there may be a large number of points in the filter. For example, if SREJ =80dB and the TWID=.01 then the number of points in the filter is 1087. The input waveform itself may be less than 1087 points long and there are edge effects, which often preclude the use of such a long filter.

The output waveform is calculated by convolving the input waveform with the impulse response of the filter:

$$output\ wfm = (input\ wfm) * h(t)$$

If the length of the original data is N and the length of the filter is NFILT, then the result of the convolution is a record N+NFILT–1 points long. See Figure C–14. From this record, (NFILT–1)/2 points are cut off each end to return a record which is N points long, the same size as the original record. However, the filter specifications are not guaranteed throughout the length of the new data record. For (NFILT–1)/2 points on either end of the new record, the data is questionable. This is due to edge effects in the convolution when the filter record is not fully within the data record.

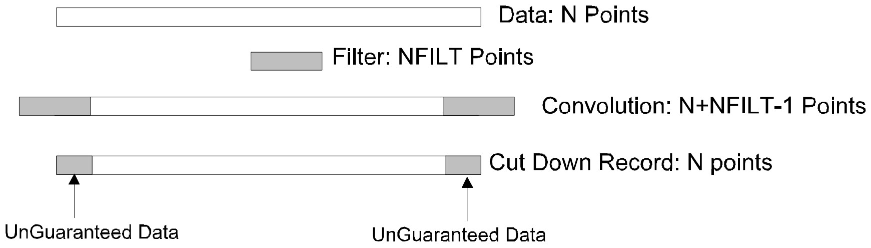


Figure C–14: Record resulting from convolving the filter impulse response with the waveform record

An example will help illustrate the edge effects. Figure C–15 shows a test signal created from a 1 V amplitude, 10 MHz sine wave and a 0.5 V, 125 MHz sine wave. This record is 500 points long, TINT=800 ps and the Nyquist frequency F_{nyq}=625 MHz. To filter out the high frequency signal, a lowpass filter was applied with LPASS = 62.5 MHz, TWID=0.1 (TWIDHZ=62.5 MHZ) and SREJ=26 dB. This resulted in a filter with β=1.51 and 53 points in the filter. Figure C–16 shows the result of applying the lowpass filter to the data. The filter did a good job of cutting out the high frequency components. To illustrate the edge effects, Figure C–17 shows a close up view of the left end of the filtered data and the 10 MHz sine signal. It is clear in that the filtered data does not initially track the source 10 MHz sine signal.

On each end of the record, (NFILT–1)/2 data points of the filtered data are not guaranteed to be within the specification of the filter. This is an undesirable situation. To limit this effect, the waveform analyzer digital filter algorithm limits the number of filter points to be a maximum of 10% of the acquisition record length. With this constraint, in the worst case condition only 5% of the data on either end of the filtered record is not guaranteed to be within the filter specification. Hence, all measurements should be on the central 90% of the data record.

To insure that your measurement does not include bad data, the waveform analyzer sets the (NFILT–1)/2 data points on either end of the filtered record to NULLs. The output waveforms of CALC blocks (which include the filter function) are always floating point numbers. NULL points are defined as

9.910000E+37 for ASCII format, and IEEE NAN (Not A Number) for REAL,32 format.

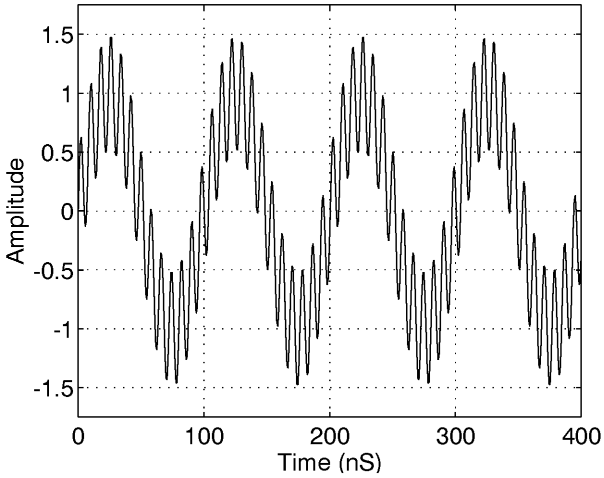


Figure C–15: Filter test signal with a 125 MHz signal modulating a 10 MHz signal

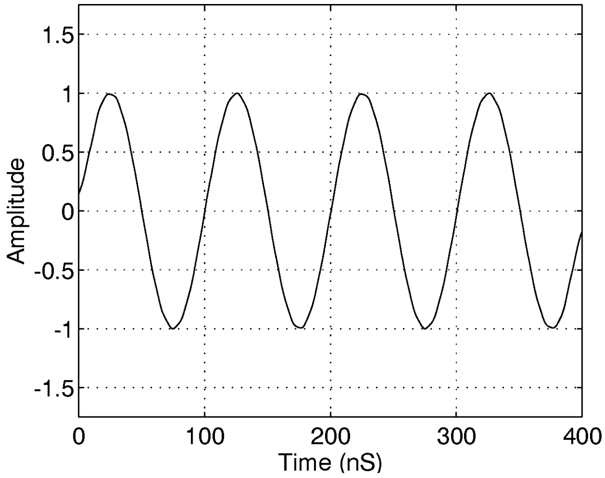


Figure C–16: Test signal after being filtered with a lowpass filter

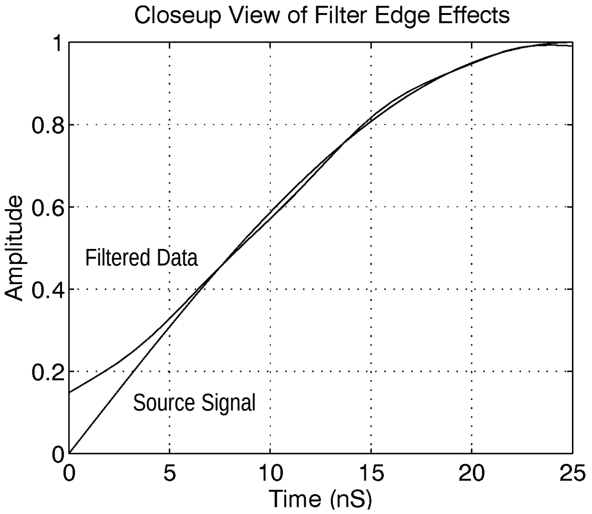


Figure C–17: View of the filtered record showing the first 5% of the filtered data

Filter Performance

Stop Band Attenuation. The stop band attenuation, or stop band rejection, is set by the SREJ parameter. The attenuation in the stop band is at least the value given by SREJ. Typically the minimum attenuation occurs at the start of the stop band. Further into the stop band the attenuation is typically several to tens of dB greater than SREJ.

Pass Band Ripple. The ripple in the pass band is not explicitly set through the filter commands. For the Kaiser window algorithm used in the waveform analyzer the pass band ripple is directly related to the stop band attenuation:

$$PassBand\ Ripple\ (dB) = 20log_{10}\left[1 + 10^{\frac{-SREJ}{20}}\right]$$

For example, if the SREJ is set to 40dB then the pass band ripple will be less than 0.0864 dB; if SREJ is set to 60dB, pass band ripple will be less than 0.00868 dB.

Filter Cutoff and Roll Off. The digital filters implemented in the waveform analyzer are different than traditional analog filters. One important distinction is that the STAR, STOP, LPAS, and HPAS frequencies are not the –3 dB cutoff frequencies for the filter but are instead –6 dB cutoff frequencies.

Analog filter designers often characterize filters by the roll off rate beyond a cutoff frequency. For example, analog filters are characterized with a roll off of 20 dB/decade, 40 dB/decade, etc. Unfortunately, these familiar analog terms do not apply to digital filters. Unlike analog filters, digital filters do not continue to drop off above a cutoff frequency. Instead, the filter response drops rapidly in the transition region and then flattens out somewhat in the stop band. In the transition region, the roll off cannot be well approximated as a constant roll off per decade of frequency (such as 40 dB/decade).

The Kaiser window filter technique does not provide a constant dB/decade roll off in the transition region. In fact, in the transition region, the Kaiser window technique only specifies that the transfer function will decrease from the pass band level to the stop band attenuation. What the Kaiser window does guarantee is the specifications in the pass band and stop band:

$$Passband\ Ripple \le 20log_{10}\left[1 + 10^{\frac{-SREJ}{20}}\right] \quad f \le LPAS - \left(\frac{TWID}{4 \cdot TINT}\right)$$

$$Stopband\ Rejection \ge SREJ \qquad f \le LPAS + \left(\frac{TWID}{4 \cdot TINT}\right)$$

Similar specifications are achieved for highpass, bandpass and notch filters. For notch filters, be sure that (STOP– START) is greater than TWIDHZ or else no guarantee is made about the attenuation in any portion of the notch region.

Group Delay. The digital filters have linear phase in the pass band. The group delay, which is the derivative of the phase, is therefore constant in the pass band.

Practically speaking, this means that if you have a signal which is made up of many frequency components, the relative phase of these frequency components are preserved in the filter.

Error Conditions

There are two main causes of errors from the digital filter code. One of the sources of error is a filter specification that generates too many filter coefficients. The other class of errors is from cutoff frequencies that violate certain constraints.

Too Many Filter Coefficients. If the stop band attenuation SREJ and/or the relative filter transition width TWID is set to too high, then the number of points required by the filter may exceed 10% of the acquired record length. Since the digital filter implementation limits the number of coefficients to 10% of the record length, waveform analyzer reports an error, and performs no filtering.

Suppose, for example, you acquire a record with 1024 points at 1 GSample/second acquisition rate. You set the lowpass filter to a cutoff frequency of 200 MHz, a stop band attenuation, SREJ, of 80 dB and relative filter transition width, TWID, of 0.05. Such a filter requires 219 points, which is more than 102 points (10% of the data record), and the following error is reported:

```
2100,"CalculateN questionable; digital filter error - filter
specs require too many coefficients"
```

When this error occurs, there are a number of things that can be done to allow the filter generation to succeed. One change you could do is to change the value for TWID (and equivalently TWIDHZ) and keep the remaining filter parameters fixed. This value of TWID is calculated from:

$$TWID = \frac{SREJ - 1.9794}{7.1785 \cdot (MaxNPTS)}$$

Where MaxNPTS in this case is set to 102. The maximum possible value for TWID is 1 (for realistic filters it is desirable to have TWID less than 0.2). If the specified SREJ is too large then the value of TWID calculated in the above expression may exceed 1. In this case, it is not possible to design a filter that meets the SREJ specifications and uses less than the maximum possible number of points.

Another change you could do is to change the value of SREJ and keep the remaining filter parameters fixed. This value of SREJ is calculated from:

$$SREJ = 7.1785 \cdot TWID \cdot (MaxNPTS) + 1.9794$$

The minimum value for SREJ is 15 dB. If the TWID was initially chosen too small then the above formula may predict a value for SREJ which is less than 15 dB. In this case, it is impossible to create a filter which meets the specification for TWID and which uses less than the maximum possible number of points.

Another change would be to acquire the data using a longer record length. Since the filter can have more points if the input data record length is larger, you can use a tighter specified filter with a longer record length. In our example, going to a record length of 4096 would have allowed the filter to operate without error.

Incorrect Cutoff Frequencies. The basic problem is that you can’t have the cutoff frequency for low or highpass filters close to 0 or Nyquist. For bandpass and notch filters, there is the additional constraint that start and stop frequencies can’t be too close to each other. The rules are as follows:

- Insure that the cutoff frequency (LPAS or HPAS) minus half the transition width is greater than 0.
- Insure that the cutoff frequency (LPAS or HPAS) plus half the transition width is less than Nyquist.

For bandpass/notch filters, insure that START minus half the transition width is greater than 0, STOP plus half the transition width is less than Nyquist, and START plus half the transition width is less than STOP minus half the transition width.

If these constraints are violated, the following error messages appear:

```
2100,"CalculateN questionable; digital filter error – lowpass
filter cutoff invalid", or
```

```
2100,"CalculateN questionable; digital filter error –
highpass filter cutoff invalid", or
```

```
2100,"CalculateN questionable; digital filter error –
bandpass/notch filter start/stop invalid"
```

Note that for both major sources of error, errors are detected and reported when the calculate block is executed, not when filter parameters are defined or when the filter expression is defined. The errors are delayed because the sample rate (which defines Nyquist) and record length (which sets the maximum number of filter coefficients) may change after you define the filter parameters.

When these errors occur, the waveform analyzer CALC block copies its input waveform to its output and performs *no* filtering.

General Guidelines

Edge effects from the filter can set up to 5% of the data on either end of the filtered record to NULL values: ASCII, 9.910000E+37 or binary, IEEE NAN. If you are going to filter the data, make sure all acquisitions have a sufficient number of points on either side of the data of interest.

If you apply a filter to a set of data and get an error, you can either specify a filter with less stringent specifications or you acquire the data a second time using a longer record length. The maximum length of the filter is 10% of the data record, so if you use a longer record you have more points available in the filter and you can use a higher specified filter.

To increase the number of points in your data, use the same sample rate but a longer record length. See the commands SWEep:POINts and SWEep:TIME to set the record length. Make sure you acquire additional points before and after the data of interest. Do not try to obtain more points in your data record by simply increasing the sampling rate. Increasing the sampling rate provides more points but it also increases the Nyquist sampling rate which can make the desired filter more difficult to achieve.

Index

Index